Simulation-optimization for fugitive interception

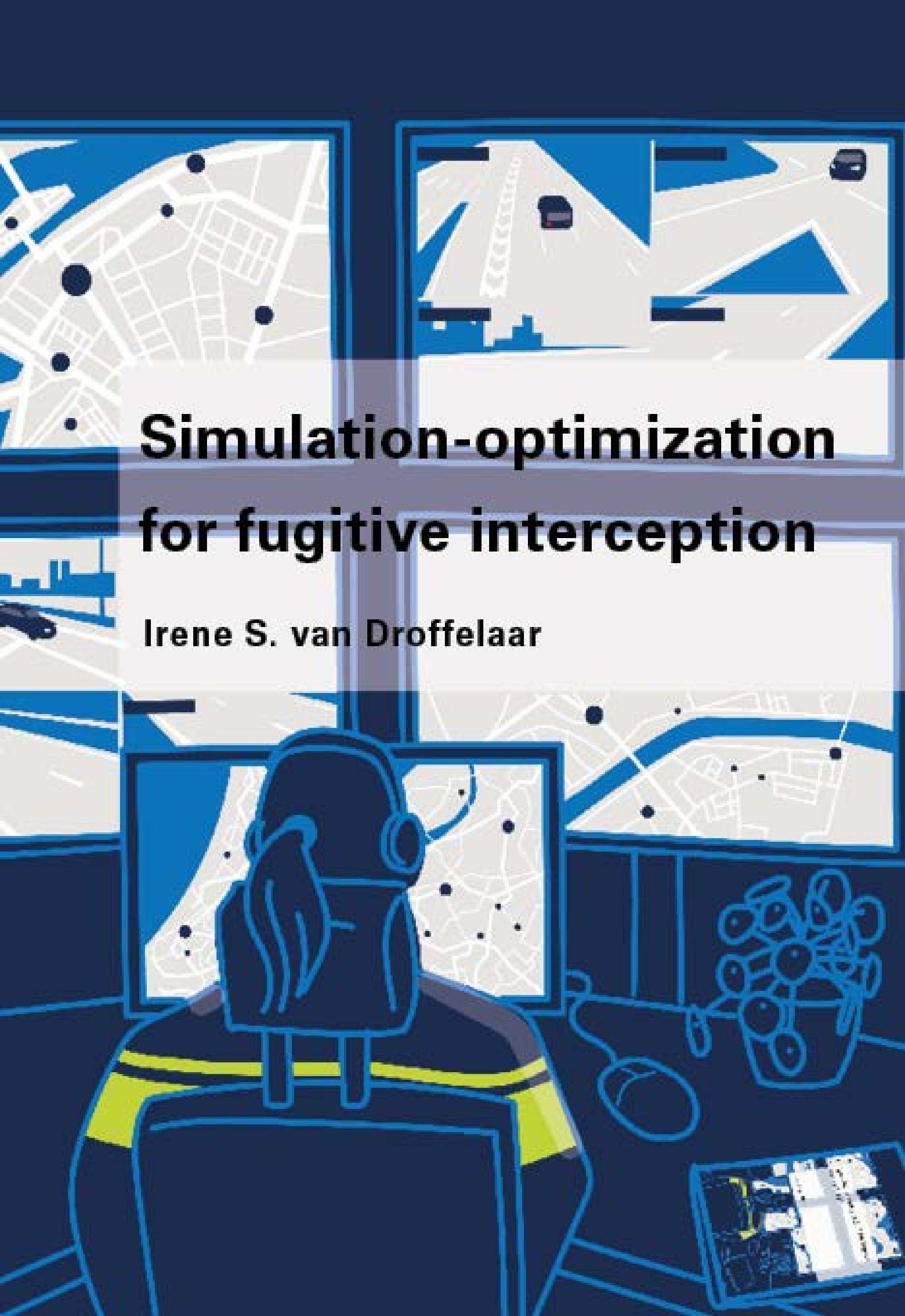
van Droffelaar, I.S.

DOI
10.4233/uuid:793d3a90-c291-4ef3-a53f-f13745f89733

Publication date
2025

Document Version
Final published version

Citation (APA)
van Droffelaar, I. S. (2025). *Simulation-optimization for fugitive interception*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:793d3a90-c291-4ef3-a53f-f13745f89733



Simulation-optimization for fugitive interception

Irene S. van Droffelaar

SIMULATION-OPTIMIZATION FOR FUGITIVE INTERCEPTION

Irene Sophia van Droffelaar

Delft University of Technology

SIMULATION-OPTIMIZATION FOR FUGITIVE INTERCEPTION

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof. dr.ir. T.H.J.J. van der Hagen;
Chair of the Board for Doctorates
to be defended publicly on
Wednesday 7 May 2025 at 15:00

by

Irene Sophia VAN DROFFELAAR

Master of Science in Engineering and Policy Analysis,
Delft University of Technology, the Netherlands.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. ir. A. Verbraeck	Delft University of Technology, *promotor*
Prof. dr. ir. J.H. Kwakkel	Delft University of Technology, *promotor*
Independent members:	
Prof. dr. S.G. Henderson	Cornell University
Prof. dr. S.J.E. Taylor	Brunel University
Prof. dr. M.E. Warnier	Delft University of Technology
Dr. C.N. van der Wal	Delft University of Technology
Prof. dr. W.W. Veeneman	Delft University of Technology, *reserve member*
Other members:	
Dr. ir. J.P. Mense,	Delft University of Technology

Dr. ir. J.P. Mense contributed significantly to the preparation of this dissertation.

The research leading to this dissertation was fully funded by the National Police Artificial Intelligence Lab (ICAI).

TRAIL Thesis Series no. T2025/5, the Netherlands Research School TRAIL

TRAIL, P.O. BOX 5017, 2600 GA Delft, The Netherlands
E-mail: info@rsTRAIL.nl

ISBN: 978-90-5584-360-2

An electronic version of this dissertation is available at https://repository.tudelft.nl

Printed by Haveka BV, the Netherlands
Cover design by Marta Ríos

What, like it's hard?

Elle Woods

CONTENTS

SUMMARY

In recent years, only 32% of reported crimes in the Netherlands resulted in the apprehension of a suspect (Moolenaar et al., 2023). Similarly, in the US, this figure is 37% for violent crimes and only 12% for property crimes (Federal Bureau of Investigation, 2022). Furthermore, 85% of arrests are red-handed, while the remaining 15% of arrests involve costly and time-intensive investigations (van Dijk et al., 2013). Increasing the number of red-handed arrests allows for more effective use of critical resources. A red-handed arrest (*heterdaad*) can happen in three ways: the arrest is made at the crime scene, during an immediate pursuit, or through a planned interception, where officers arrest the suspect while they are attempting to flee (Wetboek van Strafvordering, artikel 128, 1926). This dissertation addresses the latter: police interception. Understanding the movement patterns of fleeing suspects and suggesting proper intervention positions for police units can limit the use of police resources and increase the chance of red-handed arrests.

Search and interception of fugitives by the police on a road network is a challenging task due to the complexity of the network, the unknown whereabouts of the fugitive and uncertainty about the routes that the fugitive takes, and time pressure (Skinner & Parrey, 2019). The police control room centralists have, at most, a few minutes to decide where to position intercepting police units. Time pressure has an adverse effect on the amount of information that can be processed and, therefore, on the quality of the decision-making process (Phillips-Wren & Adya, 2020). Additionally, survivorship bias (there is only information on successful cases where the suspect was caught) and historical bias (the knowledge and intuition of officers may no longer reflect the current *modi operandi*) threaten the effectiveness of police interception.

Information technology, supported by modeling and simulation to depict the complex and stochastic decision space, can mitigate these effects by suggesting interception positions for police units. This dissertation explores simulation-optimization methods for fugitive interception, aiming to overcome barriers to using models for decision support in this context. This dissertation addresses four barriers to using models for decision support in fugitive interception: timely simulation-optimization, effective representation of the search space, simulation of fugitive behavior, and adaptation to information updates.

First, Chapter 2 proposes models and solution approaches that can encode complex behavior while optimizing the solution in real time. Simulation-optimization models are well-suited for real-time decision-support to the control room for search and interception of fugitives by Police on a road network, due to their ability to encode complex behavior while still optimizing the interception. The typical simulation-optimization configuration is simulation model optimization, where the simulation model describes the system to be optimized,

and the optimizer attempts to find the combination of decision variables that maximizes the interception probability. However, the repeated evaluation of the simulation model leads to high computation time, thus rendering it inadequate for time-constrained decision contexts. To support police interception operations in real-time, timely calculation of the solution is essential. Sequential simulation-optimization, where the simulation model, with its rich behavior, constructs (part of) the constraints of an optimization problem, could decrease the computation time. We compare the computation time for two configurations of simulation-optimization (typical simulation model optimization and sequential simulation-optimization) for various problem instances of the fugitive interception problem. We show that sequential simulation-optimization reduces the computation time of large instances of the fugitive interception case study ten-fold. This result illustrates the potential of sequential simulation-optimization to mitigate the expensive optimization of simulation models.

Second, Chapter 3 compares graph coarsening approaches to improve the timeliness and scalability of the simulation-optimization models without compromising the quality of the police interception. The number of nodes in the network, each being a crossing where routes of the fleeing suspect can split, greatly contributes to the computation time. Graph coarsening is a promising approach to reduce the complexity of the network, and therefore the computation time. We compare four graph coarsening algorithms on five road networks and assess their impact on computation time and solution quality for the fugitive interception problem. Based on the comparison, we propose and test a new method specifically for fugitive interception. This method, Search Space Representation, improves the quality of the best solutions obtained by the optimization algorithm with up to 12%, improves the reliability of the optimization to find high-quality solutions, and decreases the number of function evaluations required to obtain high-quality solutions to 5 000 - 10 000 depending on the size and complexity of the road network, which is feasible for real-time decision-making. Search Space Representation can be applied to reduce the computation time of other network-based optimization problems.

Third, this dissertation leverages psychological theory to develop behaviorally explicit models. Various conceptualizations of route choice decision-making of fleeing suspects exist. We operationalize two models of route choice and implement these in a simulation. Chapter 4 explores these different models of fugitive behavior lead to vastly different routes and, therefore, calculated interception positions. The experiments show that the different route-choice models result in different escape routes and, therefore, different calculated police interception positions. The differences are larger when the road network is complex and contains non-uniform obstacles (for example, cameras and traffic lights). In other words, the robustness of the calculated police interception positions for each model largely depends on the network topology.

Fourth, Chapter 5 compares solution approaches to handle the continuously changing decision environment, where police units and the fugitive are on the move, and new information becomes available. The police can access traffic cameras with automatic number plate recognition (ANPR) software. Additionally, they may receive calls from concerned citizens regarding abnormal or dangerous behavior. These information sources can help to narrow the search for a fleeing fugitive and, therefore, increase the probability of interception. Using the information to increase the probability of interception is not straightforward due to the unpredictability of the fleeing suspect and not knowing if, where, and when a traffic camera detects the fugitive introduce uncertainty to the problem. Moreover, there is clear path dependency: sending police units in a certain direction constrains their possible rerouting in the future. In other words, there is a trade-off between the flexibility to react to new information and the timeliness of decisions. Traditional stochastic online optimization methods, such as Periodic Re-Optimization, do not account for uncertainty or path dependency. Two promising adaptive solution approaches are Policy Tree Optimization and Direct Policy Search. However, these solution approaches have not been applied to fugitive interception, which has a rugged fitness landscape and requires the solution to be calculated in real time to be relevant to decision-makers. We examine the solution quality obtained for the fugitive interception problem with information updates within a limited number of function evaluations for four solution approaches: One-shot Optimization (the static optimization models used throughout the dissertation), Periodic Re-Optimization, Direct Policy Search, and Policy Tree Optimization. Based on the experiments, we advise using Direct Policy Search for problems that are vulnerable to lock-ins. If the interpretability of the results is critical, Direct Policy Search should be supplemented with an interpretable interface. Otherwise, we advice using Periodic Re-optimization for its flexibility and ease of implementation.

Finally, Chapter 6 evaluates the models and solution approaches developed in the previous chapters by comparing their outputs to the actual locations where police units were positioned by the control room. To apply models in the police control room to advise on interception strategies, the models need to be thoroughly validated and tested. The evaluation discussed in Chapter 6 is the first step in this process. The evaluation helps to identify strengths and limitations in the simulation-optimization approach and leads to recommendations for improvements and further research.

We highlight three main contributions of this dissertation. First, we demonstrate how sequential simulation-optimization reduces computation time compared to classical simulation model optimization. Additionally, we present a metaheuristic solution approach that identifies near-optimal solutions in a fraction of the time required for exact optimization, with the computation time increasing at a slower rate as the network size grows. Second, we identify a method for incorporating information updates — both observations and the absence of observations of the fugitive — into the interception strategy while maintaining consistency in the interception positions. Third, we show how behavioral assumptions impact the effectiveness of interception strategies. More detailed models of behavior can easily be incorporated into the proposed simulation-optimization method.

To summarize, this research provides the foundation for effective decision support to police control rooms to increase the chance of red-handed arrests.

Samenvatting

In de afgelopen jaren resulteerde slechts 32% van de geregistreerde misdrijven in Nederland in de aanhouding van een verdachte (Moolenaar e.a., 2023). Dit percentage ligt in de VS op 37% voor geweldsmisdrijven en slechts 12% voor vermogensdelicten (Federal Bureau of Investigation, 2022). Bovendien wordt 85% van de aanhoudingen op heterdaad verricht — de overige 15% van de aanhoudingen gaat gepaard met kostbare en tijdrovende onderzoeken (van Dijk e.a., 2013). Het verhogen van het aantal heterdaadaanhoudingen maakt een effectievere inzet van kritieke middelen mogelijk. Een heterdaadaanhouding kan op drie manieren plaatsvinden: de aanhouding wordt verricht op de plaats delict, tijdens een directe achtervolging, of via een geplande interceptie, waarbij de politie de verdachte aanhoudt terwijl deze probeert te vluchten (Wetboek van Strafvordering, artikel 128, 1926). Dit proefschrift richt zich op het laatste: politie-interceptie. Inzicht in de bewegingspatronen van vluchtende daders en het voorstellen van kansrijke opstelposities voor politie-eenheden kan het gebruik van politiemiddelen beperken en de kans op heterdaadaanhoudingen vergroten.

Het opsporen en onderscheppen van vluchtende daders door de politie op een wegennet is een uitdagende taak vanwege de complexiteit van het netwerk, de onzekerheid over de route die de vluchtende dader neemt, en de tijdsdruk (Skinner & Parrey, 2019). De centralisten in de meldkamer van de politie hebben hooguit enkele minuten om te beslissen waar de onderscheppende politie-eenheden moeten worden gepositioneerd. Tijdsdruk heeft een negatief effect op de hoeveelheid informatie die kan worden verwerkt en daarmee op de kwaliteit van het besluitvormingsproces (Phillips-Wren & Adya, 2020). Daarnaast bedreigen *survivor bias* (er is alleen informatie beschikbaar over succesvolle gevallen waarin de verdachte is aangehouden) en historische bias (de kennis en intuïtie van agenten weerspiegelen mogelijk niet langer de huidige *modi operandi*) de effectiviteit van interceptiestrategieën.

Informatietechnologie, ondersteund door modellering en simulatie om complexe en onvoorspelbare situatie in kaart te brengen, kan deze effecten verminderen door opstelposities voor politie-eenheden voor te stellen. Dit proefschrift onderzoekt simulatie-optimalisatiemethoden voor het onderscheppen van vluchtende daders, met als doel de inzet van modellen voor besluitvorming in deze context te verbeteren.

Dit proefschrift richt zich op vier barrières bij het gebruik van modellen ter ondersteuning van besluitvorming bij het onderscheppen van vluchtende daders: tijdige simulatie-optimalisatie, effectieve representatie van de zoekruimte, simulatie van het gedrag van vluchtende daders, en het aanpassen van de opstelposities op basis van informatie-updates.

Ten eerste stelt Hoofdstuk 2 modellen en oplossingsmethoden voor die het gedrag van de vluchter beschrijven en de opstelposities in realtime optimaliseren. Simulatie-optimalisatiemodellen zijn bijzonder geschikt om de meldkamer in realtime te ondersteunen bij het zoeken naar en onderscheppen van voortvluchtigen op een wegennet. Deze modellen kunnen complex gedrag representeren en tegelijkertijd de onderschepping optimaliseren. De typische configuratie van simulatie-optimalisatie is simulatiemodeloptimalisatie, waarbij het simulatiemodel het te optimaliseren systeem beschrijft en de optimizer probeert de combinatie van beslissingsvariabelen te vinden die de kans op onderschepping maximaliseert. Het herhaaldelijk uitvoeren van het simulatiemodel leidt echter tot een hoge rekentijd, wat het ongeschikt maakt voor besluitvorming met beperkte tijd. Om de meldkamer van de politie in realtime te ondersteunen bij het onderscheppen van vluchtende daders, is een snelle berekening van de oplossing essentieel. Sequentiële simulatie-optimalisatie, waarbij het simulatiemodel, met rijk gedrag, (een deel van) de randvoorwaarden van een optimalisatieprobleem opstelt, kan de de rekentijd aanzienlijk verkorten. We vergelijken de rekentijd voor twee configuraties van simulatieoptimalisatie (typische simulatiemodeloptimalisatie en sequentiële simulatieoptimalisatie) voor verschillende probleeminstanties van het optimalisatieprobleem. We laten zien dat sequentiële simulatieoptimalisatie de rekentijd van grote instanties van het optimalisatieprobleem met een factor tien vermindert. Dit resultaat illustreert het potentieel van sequentiële simulatieoptimalisatie om de hoge optimalisatiekosten van simulatiemodellen te beperken.

Ten tweede vergelijkt Hoofdstuk 3 *graph coarsening* algoritmen (die netwerken verkleinen door het verlagen van de granulariteit) van het weggennetwerk om de tijdigheid en schaalbaarheid van de simulatie-optimalisatie te verbeteren zonder de kwaliteit van de politie-interceptie te verlagen. Het aantal knopen in het netwerk, elk een kruising waar routes van de vluchtende verdachte zich kunnen splitsen, draagt sterk bij aan de rekentijd. *Graph coarsening* is een veelbelovende aanpak om de complexiteit van het netwerk, en daarmee de rekentijd, te verminderen. We vergelijken vier *graph coarsening* algoritmes voor vijf wegennetwerken en evalueren hun invloed op de rekentijd en oplossingskwaliteit voor het optimalisatieprobleem van het onderscheppen van vluchtende daders. Op basis van de vergelijking stellen we een nieuwe methode voor specifiek voor het onderscheppen van vluchtelingen. Deze methode, waarbij onbelangrijke delen van het weggennetwerk weg worden gelaten in de representatie van de zoekruimte, verbetert de kwaliteit van de oplossingen die door het optimalisatiealgoritme worden verkregen met maximaal 12%, verbetert de betrouwbaarheid van de optimalisatie om oplossingen van hoge kwaliteit te vinden, en vermindert het aantal functie-evaluaties dat nodig is om oplossingen van hoge kwaliteit te verkrijgen tot 5000 - 10.000 (afhankelijk van de grootte en complexiteit van het wegennetwerk), wat bruikbaar is voor real-time besluitvorming. Deze methode kan breder worden toegepast om de rekentijd van andere netwerkgebaseerde optimalisatieproblemen te verminderen.

Ten derde, maakt dit proefschrift gebruik van psychologische theorie om expliciete gedragsmodellen te ontwikkelen. Er bestaan verschillende conceptualisaties van routekeuze van vluchtende verdachten. We operationaliseren twee modellen van routekeuze en implementeren deze in een simulatiemodel. Hoofdstuk 4 onderzoekt of deze verschillende modellen van vluchtgedrag leiden tot zeer verschillende routes en dus berekende opstelposities voor de politie. De experimenten laten zien dat de verschillende routekeuzemodellen resulteren in verschillende vluchtroutes en dus verschillende opstelposities. De verschillen zijn groter als het wegennetwerk complex is en niet-uniforme obstakels (bijvoorbeeld camera's en verkeerslichten) bevat. Met andere woorden, de robuustheid van de berekende onderscheppingsposities voor elk model hangt deels af van de netwerktopologie.

Ten vierde vergelijkt Hoofdstuk 5 oplossingsmethoden om om te gaan met de continu veranderende context, waar politie-eenheden en de vluchtende verdachte in beweging zijn en nieuwe informatie beschikbaar komt. De politie heeft toegang tot verkeerscamera's met automatische nummerplaatherkenningssoftware (ANPR). Daarnaast kunnen ze telefoontjes ontvangen van bezorgde burgers over abnormaal of gevaarlijk gedrag. Deze informatiebronnen kunnen helpen om de zoektocht naar een vluchtende dader te vernauwen en zo de kans op onderschepping te vergroten. Het gebruik van de informatie om de waarschijnlijkheid van onderschepping te vergroten is niet eenvoudig omdat de vluchtende verdachte onvoorspelbaar is en omdat niet bekend is of, waar en wanneer een verkeerscamera de vluchtende verdachte detecteert. Bovendien is er padafhankelijkheid: het sturen van politie-eenheden in een bepaalde richting beperkt hun mogelijke veranderingen in opstelpositie in de toekomst. Met andere woorden, er is een afweging tussen de flexibiliteit om te reageren op nieuwe informatie en de tijdigheid van beslissingen. Traditionele stochastische online optimalisatiemethoden, zoals periodieke heroptimalisatie, houden geen rekening met onzekerheid of padafhankelijkheid. Twee veelbelovende adaptieve optimalisatietechnieken zijn *Direct Policy Search* en *Policy Tree Optimization*. Deze oplossingsmethoden zijn echter nog niet toegepast op het onderscheppen van vluchtende daders, met een onregelmatig fitnesslandschap, en waarbij de oplossing in realtime moet worden berekend om relevant te zijn voor de meldkamer. We onderzoeken de oplossingskwaliteit voor het interceptieprobleem met informatie-updates binnen een beperkt aantal functie-evaluaties voor vier optimalisatietechnieken: statische optimalisatie (die in de rest van het proefschrift worden gebruikt), periodieke heroptimalisatie, *Direct Policy Search* en *Policy Tree Optimization*. Op basis van de experimenten adviseren we om *Direct Policy Search* te gebruiken voor problemen die kwetsbaar zijn voor lock-ins. Als de interpreteerbaarheid van de resultaten kritisch is, moet *Direct Policy Search* worden aangevuld met een interpreteerbare interface. In andere gevallen adviseren we om *Periodic Re-optimization* te gebruiken vanwege de flexibiliteit en het gemakkelijke implementatieproces.

Tot slot worden in Hoofdstuk 6 de modellen en oplossingsmethoden die in de voorgaande hoofdstukken zijn ontwikkeld, geëvalueerd door hun output te vergelijken met de werkelijke locaties waar politie-eenheden door de meldkamer zijn gepositioneerd. Om de modellen toe te passen in de politiemeldkamer om advies te geven over onderscheppingsstrategieën, moeten de modellen grondig worden gevalideerd en getest. De evaluatie die in Hoofdstuk 6 wordt besproken, is de eerste stap in dit proces. De evaluatie helpt bij het identificeren van sterke en zwakke punten in de simulatie-optimalisatieaanpak en leidt tot aanbevelingen voor verbeteringen en verder onderzoek.

We onderstrepen drie bijdragen van dit proefschrift. Ten eerste laten we zien hoe sequentiële simulatie-optimalisatie de rekentijd vermindert in vergelijking met klassieke simulatiemodeloptimalisatie. Daarnaast presenteren we een *meta-heuristic* optimalisatiealgoritme die bijna-optimale oplossingen identificeert in een fractie van de tijd die nodig is voor exacte optimalisatie, waarbij de rekentijd relatief langzamer toeneemt naarmate het netwerk groter wordt. Ten tweede identificeren we een methode om informatie-updates — zowel waarnemingen als de afwezigheid van waarnemingen van de vluchtende dader — op te nemen in de interceptiestrategie met behoud van zo veel mogelijk consistentie in de opstelposities. Ten derde laten we zien hoe modellen van vluchtgedrag de effectiviteit van onderscheppingsstrategieën beïnvloeden. Gedetailleerdere gedragsmodellen kunnen eenvoudig worden opgenomen in de voorgestelde simulatie-optimalisatiemethode.

Samengevat biedt dit onderzoek de basis voor effectieve ondersteuning voor de besluitvorming van de meldkamer van de politie om de kans op heterdaadaanhoudingen te vergroten.

1

INTRODUCTION

1.1. MOTIVATION

In recent years, only 32% of reported crimes in the Netherlands resulted in the apprehension of a suspect (Moolenaar et al., 2023). Similarly, in the US, this figure is 37% for violent crimes and only 12% for property crimes (Federal Bureau of Investigation, 2022). Furthermore, 85% of arrests are red-handed, while the remaining 15% of arrests involve costly and time-intensive investigations (van Dijk et al., 2013). Increasing the number of red-handed arrests allows for more effective use of critical resources. A red-handed arrest (*heterdaad*) can happen in three ways: the arrest is made at the crime scene, during an immediate pursuit, or through a planned interception, where officers arrest the suspect while they are attempting to flee (Wetboek van Strafvordering, artikel 128, 1926). This dissertation addresses the latter: police interception. Understanding the movement patterns of fleeing suspects and suggesting proper intervention positions for police units can limit the use of police resources and increase the chance of red-handed arrests.

Search and interception of fugitives by the police on a road network is a challenging task due to the complexity of the network, the unknown whereabouts of the fugitive and uncertainty about the routes that the fugitive takes, and time pressure (Skinner & Parrey, 2019). The police control room centralists have, at most, a few minutes to decide where to position intercepting police units. Time pressure has an adverse effect on the amount of information that can be processed and, therefore, on the quality of the decision-making process (Phillips-Wren & Adya, 2020). Additionally, survivorship bias (there is only information on successful cases where the suspect was caught) and historical bias (the knowledge and intuition of officers may no longer reflect the current *modi operandi*) threaten the effectiveness of police interception.

Information technology, supported by modeling and simulation to depict the complex and stochastic decision space, can mitigate these effects by suggesting interception positions for police units. This dissertation explores simulation-optimization methods for fugitive interception, aiming to overcome barriers to using models for decision support in this context.

1.2. BACKGROUND

This dissertation addresses four barriers to using models for decision support in fugitive interception: timely simulation-optimization, effective representation of the search space, simulation of fugitive behavior, and adapting to information updates.

1.2.1. SEARCH AND INTERCEPTION OPTIMIZATION

Though, to our knowledge, no models exist of the interception of fleeing suspects, we can draw from two fields: search problems and facility location problems.

SEARCH PROBLEMS

Search problems, or pursuit-evasion problems are problems that describe strategies to locate an evader by controlling one or multiple pursuers (Chung et al., 2011). In the most basic version of the problem, introduced by Parsons (1978), k searchers and j evaders are distributed over the vertices of a graph G. All players have complete information of all other players' locations. Alternately, any subset of searchers and any subset of evaders moves to an adjacent vertex or stays at its current location. The evaders are captured when a searcher and an evader occupy the same vertex at the same time. This variant, where players take alternating turns is appropriately named the "Cops and Robbers game". Other variants, where all players move at the same time also exist. While these more accurately correspond to the real-world situation, they are much harder to solve (Fomin & Thilikos, 2008).

Search problems either aim to find the *cop number*: the number of searchers required to guarantee capture of the fugitive regardless of the fugitive's strategy (Alspach, 2004), or aim to find search strategies that guarantee capture.

The computational complexity of the problem, especially when scaling to road networks with thousands of nodes where multiple paths can be taken at each intersection, means that applying search problems for real-time decision support in the control room is, for now, infeasible.

NETWORK INTERDICTION PROBLEMS

Network Interdiction Problems are a family of optimization problems that aim to remove or monitor the nodes (or links) of a network to minimize an adversary's ability to operate or navigate efficiently. One type of network interdiction model is Maximum Flow Interdiction, minimizing the expected maximum flow that an adversary can achieve (Smith et al., 2013). Network interdiction is typically modeled as a bilevel optimization problem, where the interdictor's decisions influence the attacker's response: the interdictor decides which nodes or links to interdict (within resource constraints), and the adversary reacts by choosing the best remaining path or flow after interdiction (Cormican et al., 1998). Relevant extensions introduce incomplete information, meaning the interdictor and adversary do not know each other's preferred strategy (Smith & Song, 2020), and stochastic interdiction, accounting for multiple escape scenarios or stochastic success of interdictions (Cormican et al., 1998). Network interdiction problems have been applied to smuggling (David P. Morton & Saeger, 2007), drug trafficking (Washburn & Wood, 1995), and infrastructure protection (Murray et al., 2007).

Network interdiction problems, and bilevel optimization problems more generally (Sinha et al., 2018), are NP-hard (Hansen et al., 1992), meaning that applying network interdiction problems for real-time decision support is difficult, especially when scaling to real-world road networks with thousands of nodes and links.

FLOW INTERCEPTION PROBLEM

The Flow Interception Problem (FIP) is a special type of Facility Location Problem. Developed by Hodgson (1990) and Berman et al. (1992), the original model aims to maximize the flow intercepted by a certain number of facilities, for example, consumers who encounter at least one facility along their predetermined journeys. Applications include refueling infrastructure for alternative fuels (Shukla et al., 2011) and inspection station location problems (Li et al., 2007). Gendreau et al. (2000) extended the FIP to include a gain coefficient a_{pi} for each vertex i belonging to route r instead of implicitly relating the gain to the flow values. Tanaka and Kurita (2020) adapt the FIP to handle probabilistic interception and reward early interception of travelers.

The generalized formalization of this class of problems (Zeng et al., 2010) can be extended to fit the police chase case, where police units are placed to maximize the fraction of intercepted escape routes. Escape routes are preexisting routes or generated routes from a starting point to various destinations. However, generalized FIPs lack an important element of the police interception case: the travel constraint of the police units, meaning that police units have to arrive at their target node *before* the fugitive to intercept the fugitive.

SOLUTION APPROACHES

Good or optimal solutions to optimization problems can be obtained using exact methods, heuristics and metaheuristics. Exact optimization methods guarantee finding an optimal solution. However, many optimization problems are classified as NP-hard problems – including the Flow Interception Problem – (Borie et al., 2011), meaning they cannot be solved in polynomial time. Only small-scale instances can be solved using exact methods.

Heuristics do not guarantee an optimal solution. Heuristics are problem-specific solution methods that exploit properties of the problem to reach a solution efficiently. Therefore, they are efficient for the problem they were designed for, while being inefficient for others (Rothlauf, 2011).

A metaheuristic, on the other hand, is a generic algorithm framework that can be applied to any optimization problem. While convergence metrics can be used to track the progress of the algorithms, optimality cannot be guaranteed. Abdel-Basset et al. (2018) distinguish metaphor-based metaheuristics (for example, based on biology (evolutionary algorithms) or physics (simulated annealing)) and non-metaphor-based metaheuristics (for example, Tabu Search and Variable Neighborhood Search). State-of-the-art examples of evolutionary algorithms for multi-objective optimization problems are ϵ-NSGA-II (Deb et al., 2002) and BORG (Hadka & Reed, 2013).

1.2.2. REPRESENTATION OF THE SEARCH SPACE

The large problem size of real-world cases further threatens timely optimization. Given the large number of possibilities of escape behavior and police routing, the size of the network quickly becomes infeasibly large for real-time decision support. Graph coarsening, a technique to reduce the size of a graph while preserving essential structural properties, offers a promising approach to reducing computation time (Geisberger et al., 2008). Graph coarsening algorithms have successfully been applied to various graph-based optimization problems where reducing the number of nodes significantly improves computation time, such as routing optimization (Sanders & Schultes, 2012), the Traveling Salesman Problem (Walshaw, 2004) and graph partitioning (Chevalier & Safro, 2009).

The effectiveness of graph coarsening algorithms varies depending on the application, as the importance of the nodes and links is very case-specific. For example, coarsening for transport modeling often focuses on preserving the shortest paths (Sanders & Schultes, 2012), while coarsening for graph partitioning aims to minimize the number of edges (Chevalier & Safro, 2009; Safro et al., 2015). Pung et al. (2022) propose an algorithm that coarsens road networks using characteristics most prominent in the United States – grids and cul-de-sacs.

For fugitive interception, coarsening risks removing nodes of high importance to the fugitive interception problem, decreasing the solution quality.

1.2.3. SIMULATION OF FUGITIVE BEHAVIOR

Mathematical optimization of the interception position relies on generating escape routes for the suspect. The set of routes, and the probability distribution over the paths, determine the optimal combination of interception positions. The generated set of routes has to be complete in terms of network coverage for the mathematical optimization model will not identify the most interesting interception points.

There are various ways to conceptualize the route choices of fleeing suspects to generate a set of escape routes (Sava et al., 2016; van Gelder, 2013). Without knowledge of their underlying decision-making process, the routes may resemble a random walk through the road network. In contrast, if we had complete information on the suspect's characteristics and decisions, there would be a single deterministic route. In practice, we have incomplete information, where we have some understanding of route choices but not all, leading to a heuristic implementation of the route choice model of a fugitive.

Many theoretical studies implement a random motion for the fleeing suspect (Borie et al., 2013; Sava et al., 2016). Explicitly encoding behavior through decision rules could lead to more effective interception strategies (Simard et al., 2021). However, very little is known about the decision-making of suspects fleeing a crime.

1.2.4. ADAPTATION TO INFORMATION UPDATES

The police can access traffic cameras with automatic number plate recognition (ANPR) software. Additionally, they may receive calls from concerned citizens regarding abnormal or dangerous behavior. These information sources can help to narrow the search for a fleeing fugitive and, therefore, increase the probability of interception. Using this information to increase the probability of interception is not straightforward due to the unpredictability of the behavior of the fugitive and not knowing if, where, and when a sensor might detect the fugitive. Given the known positions of sensors that might detect the fugitive, both detections and lack of detection rule out possible escape routes. Moreover, there is clear path dependency: sending police units in a certain direction constrains their possible rerouting in the future. In other words, there is a trade-off between the flexibility to react to new information and the timeliness of decisions.

The literature presents various methods for dealing with incoming information in simulation-optimization (Henrichs et al., 2022). Yet, it is unknown which approach is most effective for fugitive interception problems. Traditional stochastic online optimization methods, such as Periodic Re-Optimization (Psaraftis, 1980), do not account for uncertainty or path dependency. On the other hand, most techniques developed for adaptive decision-making under uncertainty are developed for long-term planning problems and require ample time for analysis and intermediate input from decision-makers. Direct Policy Search (Giuliani et al., 2016; Koutsoyiannis & Economou, 2003) and Policy Tree Optimization (Herman & Giuliani, 2018) are developed for optimal control under uncertainty and may be suitable for real-time decision-making. However, timely calculation of the solution is essential to support police interception operations in real time. Direct Policy Search optimizes a policy, described by the parametrization of Radial Basis Functions, that maps the system's state (in our case, ANPR input) to control actions (Giuliani et al., 2016; Rosenstein & Barto, 2001). Policy tree optimization optimizes a binary decision tree that delineates what actions should be taken under what conditions (i.e., ANPR input) (Herman & Giuliani, 2018). In water resource management, Policy Tree Optimization yields a more interpretable output, which makes it an interesting algorithm to evaluate for fugitive interception. Interpretably presenting interception strategies to control room centralists could be valuable for supporting timely and transparent decisions in fugitive interception.

1.3. RESEARCH QUESTIONS

This research aims to identify, develop, and evaluate methods to identify effective fugitive interceptions. Four sub-research questions address the challenges in reaching this goal. Each research question aims to improve the effectiveness of the interception while preserving the timeliness of the calculated solutions. Figure 1.1 presents a graphical overview of the research challenges addressed by each research question.

1. *How to formalize fugitive interception?* Models of fugitive interception are not of sufficient maturity to apply in a control room. The networks that models are applied to are simple (not a road network, but instead lines and circles and trees) and they cannot easily be extended due to the constrained computation time for real-time decision support. This research question aims to develop models and solution approaches that can encode complex behavior while optimizing the solution in real time.

2. *How to leverage graph coarsening to improve the timeliness of simulation-optimization for fugitive interception?* This research question aims to develop graph coarsening approaches for fugitive interception to improve the timeliness and scalability of the aforementioned approaches without compromising the quality of the police interception.

3. *How to generate an ensemble of realistic fugitive escape routes?* Escape routes are often generated using random walk models. Instead, this research question aims to leverage psychological theory to develop behaviorally explicit models. This research question explores whether these models lead to vastly different routes and, therefore, calculated interception positions.

4. *How to use incoming information to increase the probability of interception?* This question aims to find a solution approach to handle the continuously changing decision environment, where police units and the fugitive are on the move, and new information becomes available, while the effectiveness of the implementation of a solution is dependent on its timeliness.

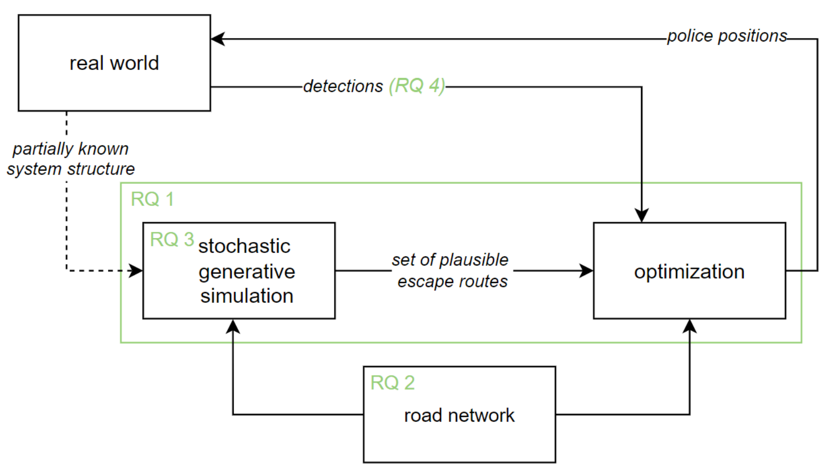


Figure 1.1: Schematic overview of the simulation-optimization framework used in this thesis. The components addressed by each research question are indicated in green.

1.4. OUTLINE

Each research question is addressed in a separate chapter. Each chapter consists of a self-contained journal or conference paper, causing some overlap in the introduction and method sections.

Chapter 2 introduces the simulation-optimization framework used throughout this dissertation. The method is supported by a literature review of simulation-optimization configurations and computational experiments comparing various approaches to optimizing fugitive interception. Chapter 3 presents a comparison of four graph coarsening algorithms to improve the tractability of the simulation-optimization problem, and proposes a graph coarsening approach for fugitive interception. Chapter 4 dives into the simulation of the escape routes of the fleeing fugitive. While the previous chapter uses a random walk model to generate escape routes, this chapter compares these to simulation models informed by criminological and route-choice literature. Chapter 5 presents a quantitative comparison of adaptive and online optimization approaches for fugitive interception. Chapter 6 evaluates the approaches presented in this dissertation for three case studies. The final chapters reflect on the research and address the research questions based on the content discussed in Chapters 2-6.

2

SIMULATION-OPTIMIZATION CONFIGURATIONS FOR REAL-TIME DECISION-MAKING IN FUGITIVE INTERCEPTION

This chapter lays the foundation for the rest of this dissertation by developing and demonstrating the feasibility of simulation-optimization for real-time decision support for fugitive interception. The remaining chapters will utilize the simulation-optimization framework presented and tested in this chapter.

This chapter is published as: Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). Simulation–optimization configurations for real-time decision-making in fugitive interception. *Simulation Modelling Practice and Theory, 133,* 102923.

The code and data associated with this chapter are available at: doi:10.4121/5f3a6a70-377b-42eb-9f46-5fd1141bed78

2

ABSTRACT

Simulation-optimization models are well-suited for real-time decision-support to the control room for search and interception of fugitives by Police on a road network, due to their ability to encode complex behavior while still optimizing the interception.

The typical simulation-optimization configuration is simulation model optimization, where the simulation model describes the system to be optimized, and the optimizer attempts to find the combination of decision variables that maximizes the interception probability. However, the repeated evaluation of the simulation model leads to high computation time, thus rendering it inadequate for time-constrained decision contexts. To support police interception operations in real-time, timely calculation of the solution is essential.Sequential simulation-optimization, where the simulation model, with its rich behavior, constructs (part of) the constraints of an optimization problem, could decrease the computation time.

We compare the computation time for two configurations of simulation-optimization (typical simulation model optimization and sequential simulation-optimization) for various problem instances of the fugitive interception problem. We show that sequential simulation-optimization reduces the computation time of large instances of the fugitive interception case study ten-fold. This result illustrates the potential of sequential simulation-optimization to mitigate the expensive optimization of simulation models.

2.1. INTRODUCTION

Search and interception of fugitives by the police on a road network is a challenging task due to the complexity of the network, the unknown whereabouts of the fugitive and uncertainty about the routes that the fugitive takes, the stressful decision-making context (Phillips-Wren & Adya, 2020), and time pressure (Skinner & Parrey, 2019). Both stress and time pressure have an adverse effect on the amount of information that can be processed and, therefore, on the quality of the decision-making process (Skinner & Parrey, 2019). Information technology, supported by modeling and simulation to depict the complex and stochastic decision space, can mitigate these effects by suggesting interception positions for police units. Related search and interception routing problems are solved using a variety of approaches. For example: general graph search (Alspach, 2004), continuous space search using mobile robotics (Chung et al., 2011), missile interception (Raap et al., 2019), and search and rescue (Sava et al., 2016), or more specifically, finding the lost MH370 (Ivić et al., 2020). Simulation-optimization models, in particular, seem suitable to solve the fugitive interception problem due to their ability to encode complex behavior and solve for good interception routes.

To support police interception operations in real-time, timely calculation of the solution is essential (van Dijk et al., 2013). Given the complexity of the problem, caused by a large number of edges in a road network, the uncertainty in the behavior of the fugitive, and the degrees of freedom of the police units, solving a typical simulation-optimization configuration in real-time is infeasible. Typically, the computation time of simulation-optimization is improved by increasing computation power and algorithm efficiency. For larger networks, however, the computation time of the fugitive interception problem would still be too large for using simulation-optimization in a real-world context where a solution is needed in less than a minute. A promising alternative is to combine simulation and optimization differently than in classical simulation-optimization, such that the number of times the simulation model has to run is drastically reduced (Figueira & Almada-Lobo, 2014).

A different way of combining simulation and optimization is sequential simulation-optimization, where the simulation constructs (part of) the constraints of an optimization problem. This paper provides an extension to the taxonomy of simulation-optimization configurations, presents and researches sequential simulation-optimization, and provides a quantitative analysis of the real-time performance of classical simulation-optimization compared to sequential simulation-optimization. We apply the comparison to a fugitive interception problem to two case studies: a 2D grid and a city road network. Thus, we show the potential of sequential simulation-optimization to mitigate the expensive optimization of simulation models.

Section 2.2 outlines the background literature on simulation-optimization. Section 2.3 describes the methods used in the paper, including a description of the case study, the models, and the optimization algorithms. The subsequent sections detail the obtained results, specifically on a grid network (Section 2.4.1) and a real-world road network (Section 2.4.2). Possible threats to the validity of the results are discussed in Section 5, and we share our conclusions in Section 6.

2.2. SIMULATION-OPTIMIZATION

Two paradigms of prescriptive analytics are simulation and optimization. Simulation answers 'what-if' questions about a system: what is the system response given a set of values for the decision variables? In contrast, optimization aims to answer 'how-to' questions: how to maximize or minimize the system response by choosing the optimal values for the decision variables? (Fu, 2015; Shannon, 1998) Simulation-optimization combines the two. Despite broad applicability, simulation-optimization is still less popular than pure simulation or optimization studies (Tordecilla et al., 2021).

The terms 'optimization via simulation' (Fu, 1994; Hong et al., 2015), or 'simulation for optimization' (Fu, 2002) are often conflated with the term simulation-optimization. They describe the application of simulation methods for solving optimization problems - not the combination of simulation and optimization models. Well-known examples are simulated annealing and ant colony optimization.

This section describes the related literature in real-time simulation-optimization and simulation-optimization configurations. In the latter subsection, we dive deeper into the configurations and provide a synthesis of concepts that add to the broader understanding of simulation-optimization.

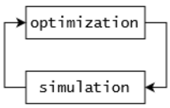
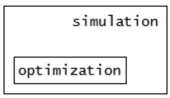
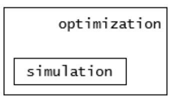
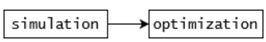
2.2.1. REAL-TIME SIMULATION-OPTIMIZATION

To be useful for real-time decision making, the timely calculation of the optimal solution is essential. In classical simulation model optimization, the simulation model is evaluated for each set of input parameter values that constitute a candidate solution determined by the optimizer. In this setup, two factors determine the computation time: (1) the number of function evaluations (*i.e.,* a single run of the simulation model) needed by the optimizer to find the optimal solution and (2) the computation time per function evaluation. The first is dependent on the efficiency of the optimizer. Since discrete simulation (DEVS) typically yields a rugged fitness landscape, the computation time of the optimization is high (Maier et al., 2019). Improvement of optimization algorithms for simulation-optimization is an active field of research with extensive literature (Amaran et al., 2016). Review articles over time are provided by, among others, (Amaran et al., 2016; Andradóttir, 1998; Carson & Maria, 1997; Fu et al., 2005; Garrison & Petty, 2019). The popularity of Digital Twins to support decision-making is increasing the need for approaches for timely simulation-optimization, and, consequently, for research done in the field (Sharma et al., 2022). Recently, the focus has shifted towards finite time performance rather than asymptotic performance, where the optimizer has reached convergence, as exemplified in (Dong et al., 2017; Ghadimi & Lan, 2015; Henderson, 2021). This development is relevant for real-time decision making, because the focus shifts to obtaining timely 'as-good-as-feasible' solutions rather than 'as-good-as-possible' solutions in as much time as the computation budget allows. For example, De Armas et al. (de Armas et al., 2017) solve the uncapacitated facility location problem for telecommunications in real time using a tailor-made simulation-based metaheuristic. However, even with improving and tailoring the optimization approach, thousands of simulation runs must be completed in the optimization search for larger problems. For complex simulation models, this leads to infeasibly high computation times for real-time simulation. The second factor, computation time per function evaluation - running the simulation model - is often not reducible due to the complexity inherent to the problem. In some cases, a solution is to develop surrogate models - or metamodels - that describe the input-output relations of the simulation model and are computationally cheaper to evaluate (Kleijnen & Sargent, 2000). However, this method requires an initial time investment to fit the surrogate model to the complex and stochastic fitness landscape of the simulation model, and information is lost in the process.

2.2.2. SIMULATION-OPTIMIZATION CONFIGURATIONS

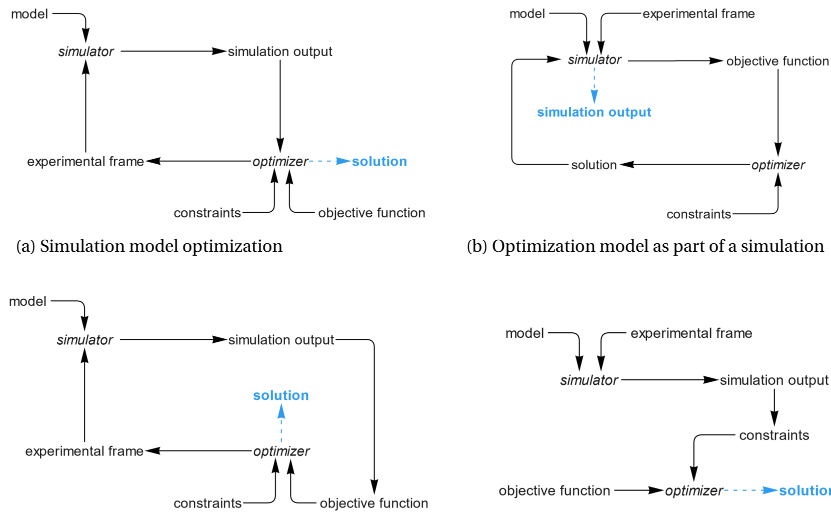
The term 'simulation-optimization' is ambiguous, with most articles describing the optimization of a simulation model (the top configuration in Table 2.1). Yet, simulation and optimization can be coupled in various configurations. Figueira and Almada-Lobo (2014) attempt to reduce the ambiguity by presenting a taxonomy for simulation-optimization. They distinguish four dimensions: simulation purpose, hierarchical structure, search method, and search scheme. Table 2.1 summarizes the hierarchical structure dimension, which this paper examines further.

Table 2.1: A tabular overview of the four simulation-optimization configurations

	Configuration	Goal	Type of output	Iterations
(a)		Optimize a simulation model	Optimal solution	Optimization: > 1 Simulation: > 1
(b)		Implement part(s) of a simulation model as optimization	Model output	Optimization: ≥ 1 Simulation: 1
(c)		Implement part(s) of an optimization model as simulation	Optimal solution	Optimization: 1 Simulation: ≥ 1
(d)		Construct constraints of an optimization model through simulation	Optimal solution	Optimization: 1 Simulation: 1

We examine the connections between simulation and optimization in each of the configurations. An optimization model consists of constraints, an objective function, and an optimizer (Nocedal & Wright, 2006). The objective function is a function of model variables that should be maximized or minimized; constraints define feasibility by imposing limitations on model variables and parameters; the optimizer, or solver, is the algorithm that finds the optimal solution. Likewise, following the Discrete Event System Specification modeling framework (Zeigler et al., 2000), a simulation model consists of an experimental frame, a model, and a simulator. The experimental frame provides the input arguments: the conditions under which the system is experimented with; the model describes the logic of the simulation; the simulator executes the model. In Figure 2.1, we use these frameworks (Nocedal & Wright, 2006; Zeigler et al., 2000) to specify the simulation and optimization components in the simulation optimization taxonomy (Figueira & Almada-Lobo, 2014). In each configuration, the model is assumed to be fixed. Changes to the model are passed through the experimental frame as changes to parameters or decision variables. Changes to the model itself would introduce structural uncertainty, which is outside the scope of this paper.

The following paragraphs describe the four configurations in more detail and provide examples. For each configuration, we discuss the purpose of the simulation and optimization components, which components are endogenous/exogenous, how stochasticity in the simulation model is managed, and an illustrative example from the literature.



(a) Simulation model optimization

(b) Optimization model as part of a simulation

(c) Simulation model as part of an optimization

(d) Sequential simulation-optimization

Figure 2.1: Configurations of combining simulation and optimization, specified using terminology from the Discrete Event System Specification modeling framework (Zeigler et al., 2000). The blue dashed arrow indicates the entity that delivers the final output to the model user.

Configuration (a): Simulation Model Optimization The simulation model describes the system to be optimized, and the optimizer attempts to find the combination of decision variables X that minimizes or maximizes the objective value $f(X)$ where f is the transformation of input to output by the simulation model and $Y = f(X)$ are the model output variables (Fu, 1994). In each iteration, the optimization and the simulation are both executed (Figueira & Almada-Lobo, 2014). For the simulation, this means that the model is run to completion. If the simulation model is stochastic, the simulation model's output is a summary statistic over multiple replications. For optimization, the optimizer receives the simulation model's output for the previous set of decision variables and determines a new set of decision variables to evaluate. The decision variables are passed to the simulator through the experimental frame.

Xi et al. (2013) describe a real-world example using an extensive simulation model of transportation behavior in the central Ohio region. The optimal locations of electric vehicle charging stations are determined to maximize the service rate. Many more examples are described in the literature, with applications in, among others, transport, logistics, and health care (Amaran et al., 2016). Extensions of simulation model optimization are, for example, robust optimization and model calibration, where additional functions are applied to the outcomes of the simulation model (van Schilt et al., 2022; Wigan, 1972). In robust optimization, the objective function is based on the robustness of the output given a set of values for the decision variables. Robustness is often defined as the variance of the output for a set of simulation runs with different input parameter values, but there is a broader array of robustness metrics available (see McPhail et al. (2018)). Simulation model calibration minimizes the distance between the output of the simulation model and the associated observed value (Wigan, 1972).

Configuration (b): Optimization Model as Part of a Simulation The optimization model describes a process within the simulation model, representing something that is also optimized in the system under study. In each iteration of the simulation, one or more complete optimization runs are performed (Figueira & Almada-Lobo, 2014). The simulator calls the optimization model, which calculates the optimal solution given the current states of the simulation, the constraints, and the objective function. The optimizer's output is endogenous and used in the remainder of the simulation run. Compared to configuration A, the objective function becomes endogenous. The experimental frame is exogenous and is determined by the model user. If the simulation model is stochastic, this entire process should be executed for multiple replications.

An illustrative example is described by van Barneveld et al. (2016), where ambulance operations are examined. An optimization model dynamically determines the re-locations of ambulances to anticipate demand each time step of the simulation model. Many agent-based models also fall into this category: each agent optimizes its own behavior, and we observe the emergent behavior on the model level.

Configuration (c): Simulation Model as Part of an Optimization The simulation model describes a process within an optimization model, which cannot or should not be characterized by constraints in the optimization model (Azadivar, 1999). Similar to configuration B, the objective function is endogenous. In contrast, the simulation output is endogenous, and the simulation-optimization output is the optimizer's solution. The optimizer provides the input for the simulation model. The simulation output contributes to a part of the objective function. The simulation typically introduces stochasticity in the optimization. Many replications should be run to obtain output with a small confidence interval since classical optimization methods cannot handle stochasticity. In each iteration of the optimization model, one or multiple simulation runs are completed (Figueira & Almada-Lobo, 2014).

An example is using a simulation model as a more realistic representation of a queue, in contrast to a simple but unrealistic, first-in-first-out (or other simple optimization-based) queuing system. Azadivar (1999) provides the example of a resource allocation model, where a simulation model of the inventory represents the stochasticity in lead time and inventory.

Configuration (d): Sequential Simulation-Optimization The simulation and optimization modules run sequentially, where the simulation experiment runs only once (Figueira & Almada-Lobo, 2014). If the simulation model is stochastic, this simulation experiment should consist of multiple replications to obtain the simulation output. Table 2.1 and Figure 2.1d depict simulation ⟶, but the simulation and optimization components could be connected in either order. In simulation ⟶ optimization, the simulation output forms (a part of) the constraint set of the optimization model. The constraints to the optimization problem are endogenous. Similar to configurations a) and c), the final output is the optimal solution. In contrast, both the simulation and optimization components are only run once, suggesting a major potential improvement in efficiency. In optimization ⟶ simulation, the optimization model provides a configuration or schedule to be used in the simulation. The final output is the simulation output, similar to configuration b).[1]

An illustrative example of sequential simulation-optimization is provided by Gülpınar et al. (2004), where simulation is used to construct scenario trees that are subsequently used in a financial portfolio optimization problem.

2.3. METHOD

In this section, we first describe the case study, independent of the specific implementations. Second, we explain the implementation of simulation model optimization (Configuration (a) from Table 2.1). Next, we present the sequential simulation-optimization formalization (Configuration (d) from Table 2.1). Finally, we discuss the solution approaches and methods for comparison.

2.3.1. CASE STUDY: FUGITIVE INTERCEPTION

We use the positioning of police units to maximize the probability of intercepting a fleeing fugitive on a road network as a case to examine the effect of simulation-optimization configurations on the computation time. The problem is modeled from the time of the incident until the fugitive is either intercepted or has escaped. Police units have no knowledge of the fugitive's whereabouts, so they have to move to a vertex in the network where the probability of intercepting is highest, *e.g.*, a chokepoint in the network where many routes pass through.

RELATED OPTIMIZATION PROBLEMS

Search problems, and more specifically, interception problems, describe related problems. A search problem related to the case study addressed in this paper optimizes the routes of searchers to maximize the probability of finding a target or to minimize the time to detect a target (Alspach, 2004). However, optimizing an

[1] A related optimization approach is sample average approximation (SAA), which was developed to solve large-scale stochastic programming problems. Stochastic programming problems maximize the expected value of the objective function, considering different possible realizations of uncertain parameters. SAA approximates the expected value of the objective function using a finite sample of scenarios, which allows for the transformation of a stochastic problem into a deterministic one by considering a representative subset of all possible scenarios. Monte Carlo simulation is a common technique within SAA to generate a set of scenarios (Kim et al., 2015; Shapiro, 2003).

action for each time step quickly becomes untractable with larger networks and longer time horizons.

Primarily applied in the field of robotics, these problems are often modeled in continuous spaces or on grids (Chung et al., 2011). In theoretical mathematical exploration, problems are solved for different graph topologies, such as grids, circular graphs, trees, and random graphs. On various graph topologies, these problems are proven to be pseudo-P to strongly NP-complete, depending on the specific problem formulation and properties of the graph (Borie et al., 2011). Due to the complexity of these optimization problems, the problem instances that are studied are typically very small. To our knowledge, there are no applications in real-world street graphs.

Problems with both stationary and moving targets have been addressed. Stationary targets may be placed randomly or adversarially to evade capture. Similarly, moving targets may be adversarial or non-reactive. Non-reactive moving targets are generally modeled as random walks (Chung et al., 2011). Extensive efforts have been made to analyze and model the behavior of lost persons (Hashimoto et al., 2022; Koester, 2008; Sava et al., 2016). However, these behaviorally rich models have not been integrated with search optimization problems, and models of criminal routing behavior have not been published.

Another related area of research describes *flow interception problems*, a special type of Facility Location Problem. Developed by Hodgson (1990) and Berman et al. (1992), the original model aims to maximize the flow intercepted by a certain number of facilities. For example, these models are used to maximize the number of consumers who encounter at least one facility on their path. Gendreau et al. (2000) extended the FIP to include a gain coefficient a_{rv} for each vertex v belonging to route r instead of implicitly relating the gain to the flow values. Tanaka and Kurita (2020) adapt the FIP to handle probabilistic interception and reward early interception of travelers. The generic Flow Interception Problem is NP-hard, meaning it cannot be solved in polynomial time (Berman et al., 1992).

MODELING CHOICES

We choose to model the problem as a variation on the Flow Interception Problem, since it is expected to be a more tractable problem to solve in real time. Instead of optimizing a position for each police unit for each time step, we optimize the target position for each police unit.

We simulate the routing behavior of the fugitive on a graph as a random walk starting at the location of the incident. At each intersection, the fugitive chooses the next vertex to travel to, which is a stochastic process where each neighboring vertex has equal probability. The fugitive does not turn around unless the vertex only has one neighboring vertex (i.e., a dead end). Each instance of this simulation generates an element r in R, consisting of $|R|$ fugitive routes. Table 2.2 specifies the model variables and parameters.

$$\phi_{r,t+1} = \mathit{Uniform}(\mathit{Neighbors}(\phi_{r,t})) \qquad \forall r \in R \quad \forall t \in T \tag{2.1}$$

2

Table 2.2: Notation of parameters and decision variables. Simulation model optimization and sequential simulation-optimization require different forms of $\pi_{u(,v)}$ and $\phi_{r(,v),t}$. For completeness, both are included in this table.

Decision variables	
$z_r \in \{0,1\}$	binary variable for the interception of route r
$\pi_u \in V$	target vertex of police unit u
$\pi_{u,v} \in \{0,1\}$	binary parameter for the target vertex v of police unit u
Parameters	
$V = \{v\}$	set of vertices
$R = \{r\}$	set of fugitive routes
$U = \{u\}$	set of police units
$T = \{t\}$	ordered index set of time steps
$\phi_{r,t} \in V$	vertex of fugitive route r at time t
$\phi_{r,v,t} = \{0,1\}$	binary parameter for the presence of fugitive route r at vertex v at time step t
$\tau_{u,t} \in V$	vertex of police unit u at time t
$\tau_{u,v,t} = \{0,1\}$	binary parameter for the reachability of vertex v by police unit u at time t

The travel time between two vertices i and j is determined by the length of the edge ($length_{i,j}$) and the maximum allowed speed on that edge ($Vmax_{i,j}$) (eq 2.2).

$$travel\ time_{i,j} = \frac{length_{i,j}}{Vmax_{i,j}} \tag{2.2}$$

We optimize the positioning of the police units ($\pi_{u,v}$) to jointly maximize the number of intercepted fugitive routes (z_r). The police units drive the shortest route from their position at the time of the incident to their respective target vertex. The police intercept a fugitive route r if a police unit has arrived at its target vertex (π_u), and the fugitive and the police unit are at the same vertex at the same time. The police units stay at their target vertex. Therefore, if a fugitive's route crosses that vertex at a later time step, it also results in an interception.

MODEL DESCRIPTION: SIMULATION MODEL OPTIMIZATION

The implementation of simulation model optimization consists of a simulation model ($f(\pi_u;\phi_{r,t},\tau_{u,t})$) that describes the movement of the fugitive, the movement of the police units, and the interception process given the starting and target nodes of the police units ($\tau_{u,t}$, π_u) and the fugitive routes ($\phi_{r,t}$). The simulation model outputs the number of intercepted fugitive routes (z_r). To account for stochasticity in the behavior of the fugitive, the model contains 500 instances of the fugitive model entity. These escape routes are the same for each function evaluation. The target vertices of the police units are optimized to maximize the number of intercepted fugitive routes (eq 2.4).

$$\text{Maximize:} \quad Z = \sum_{r\in R} z_r \tag{2.3}$$

$$\text{Subject to:} \quad z_r = f(\pi_u;\phi_{r,t},\tau_{u,t}) \tag{2.4}$$

The simulation model is implemented in pyDSOL, a Python implementation of the Distributed Simulation Object Library (DSOL) simulation library (Jacobs, 2005). [2,3]

MODEL DESCRIPTION: SEQUENTIAL SIMULATION-OPTIMIZATION

The implementation of sequential simulation model optimization consists of a simulation model that describes the movement of the fugitive and a separate optimization model that determines the routing of the police units. The simulation model is run 500 times to generate an ensemble of plausible fugitive routes and to construct $\phi_{r,v,t}$. The optimization problem is formulated as a Flow Interception Problem (Berman et al., 1992; Hodgson, 1990) with a time constraint on interception, determined by the initial position and speed of the police units.

Analogous to the simulation model formalization in the previous section, the decision variables of the optimization problem are the police unit positions $\pi_{u,v}$ and the intercepted routes z_r. Given the decision variables and parameters outlined in Table 2.2, the optimization problem is defined as follows:

$$\text{Maximize: } Z = \sum_{r\in R} z_r \tag{2.5}$$

$$\text{Subject to: } \sum_{v\in V} \pi_{u,v} == 1 \qquad \forall u \in U \tag{2.6}$$

$$z_r = \min\Big(1, \sum_{u\in U}\sum_{t\in T}\sum_{v\in V} \phi_{r,v,t}\cdot\pi_{u,v}\cdot\tau_{u,v,t}\Big) \qquad \forall r \in R \tag{2.7}$$

The objective function of the optimization (2.5) describes the maximization of the number of intercepted routes at vertices $\pi_{u,v}$. Furthermore, $\tau_{u,v,t}$ is a given for any starting point of a police unit and can be pre-loaded. Constraint 2.6 ensures that only one position is chosen for each police unit. Constraint 2.7 ensures that a route is intercepted if a police unit is placed at any vertex on a route r and the police unit can reach vertex v at time t. If a route contains more than one police unit, it will be counted in the objective function once since z_r is a binary variable. If no vertex in route r contains a police unit, the variable z_r equals 0 (i.e., not intercepted). z_r equals 1 (i.e., is intercepted) if a police unit is present on at least one vertex in route r, (Boccia et al., 2009).

2.3.2. SOLUTION APPROACHES

EXACT OPTIMIZATION

Exact optimization methods guarantee to find an optimal solution. However, the Flow Interception Problem is NP-complete (Borie et al., 2011), meaning they cannot be solved in polynomial time. Only small-scale instances can be solved using exact methods, threatening the real-time applicability of exact optimization methods. Regardless, many different commercial and open-source exact solvers apply (a mixture of) approaches that exploit common characteristics of these classes of

[2]pyDSOL core: https://github.com/averbraeck/pydsol-core
[3]pyDSOL model: https://github.com/imvs95/pydsol-model

problems to find the optimal solution efficiently. We choose the Coin-OR branch-and-cut open-source solver (Ralphs, 2022) for its applicability to Mixed-Integer Problems and open-source availability.

We use the true optimum found by the exact solver to assess the convergence of the metaheuristic optimization algorithm.

METAHEURISTIC OPTIMIZATION

Heuristics are problem-specific solution methods that exploit the properties of the problem to reach a solution efficiently but do not guarantee an optimal solution. Therefore, they are effective for the problem they were designed for while being inefficient for others (Rothlauf, 2011). Conversely, a metaheuristic is a generic algorithm that can be applied to any optimization problem. While convergence metrics can be used to track the algorithm's progress, optimality cannot be guaranteed. Abdel-Basset et al. (2018) distinguish metaphor-based metaheuristics (for example, based on biology (*e.g.*, evolutionary algorithms) or physics (*e.g.*, simulated annealing)) and non-metaphor-based metaheuristics (*e.g.*, Tabu Search and Variable Neighborhood Search). Genetic algorithms - a subset of evolutionary algorithms - generally perform well on combinatorial optimization problems with complex interactions between decision variables (Mühlenbein et al., 1988; Tolk, 2022; Torres & Khuri, 2001). State-of-the-art examples of evolutionary algorithms for multi-objective optimization problems are ϵ-NSGA-II (Deb et al., 2002) and Borg (Hadka & Reed, 2013).

We choose a simple genetic algorithm supplemented with the auto-adaptive framework from Borg, which co-evolves the probabilities of the evolutionary operators used for population adaptation based on their relative success in finding fitter offspring. This means that the algorithm optimizes the probability of each operator being used during the optimization, speeding up convergence by leveraging each operator when performing best. The operators used are 1) Simulated Binary Crossover (SBX), 2) Differential Evolution (DE), 3) Parent-Centric Crossover (PCX), 4) Simplex Crossover (SPX), 5) Unimodal Normal Distribution Crossover (UNDX), and 6) Uniform Mutation (UM) applied with probability (Hadka & Reed, 2013). At the initialization of the algorithm, each operator has equal probability. We use the default settings for Borg, as presented in Table 2.3. Further research should systematically compare various suitable optimization algorithms, for example, using a testbed like Eckman et al. (2023).

The solutions of the metaheuristic are scaled to the solution found by the exact optimization approach. Therefore, a scaled score of 1 means that the metaheuristic has found the best possible solution, not that the solution intercepts all fugitive routes.

2.3.3. SEARCH SPACE REPRESENTATION

Following Bode et al. (2019), three search space representation measures are implemented to speed up convergence: a linear index representation of the search space and consecutive filtering and sorting of the possible values for the decision variables.

Table 2.3: Default Settings of the Borg MOEA (Hadka & Reed, 2013). For the PM rate and UM rate, L is the number of decision variables.

Parameter	Value	Parameter	Value
PM rate	1/L	PCX nr. of parents	10
PM distribution index	20	PCX nr. of offspring	2
SBX rate	1	PCX eta	0.1
SBX distribution index	15	PCX Zeta	0.1
DE crossover rate	0.1	UNDX nr. of parents	10
DE step size	0.5	UNDX nr. of offspring	2
UM rate	1/L	UNDX eta	0.1
SPX nr. of parents	10	UNDX zeta	0.1
SPX nr. of offspring	2	Population size	100
SPX epsilon	0.3	Offspring size	200
		Logging frequency	200 nfe

The standard way to represent the set of possible target vertices for the police units is the *binary representation*, where each combination of police unit u and vertex v is a binary variable $\pi_{u,v}$. Bode et al. (2019) signal that evolutionary algorithms have difficulty traversing the search space due to the large set of possible combinations and strong interdependency of decision variables. Therefore, Bode et al. (2019) propose the *linear index representation*, where the decision variables are linear indices that point to the target vertex for each of the police units. Therefore, the number of decision variables only depends on the number of police units to be positioned and not also on the number of possible target vertices.

Secondly, Bode et al. (2019) suggest sorting the indices of the linear index representation so that proximity in the search space is more related to proximity in the objective space. Therefore, we sort the possible target vertices for the police units on their proximity to the starting vertex of the fugitive.

Lastly, we reduce the search space size by filtering the vertices that cannot be reached within the planning horizon by the fugitive or the respective police unit. These vertices do not contribute to increasing the objective value and do not need to be considered. This filtering considerably decreases the set of possible values for each decision variable and, therefore, the number of permutations, especially for instances with a high number of police units (Table 2.4).

2.3.4. DESIGN OF EXPERIMENTS

We examine the effect of simulation-optimization configurations on the computation time for varying problem sizes. Specifically, we vary two parameters: the number of vertices in the graph and the number of police units to be positioned and record the computation time. The number of police units determines the number of decision variables and is, therefore, expected to have an effect on the computation time. Preliminary experiments demonstrated that the computation time is especially sensitive to the number of vertices in the network (van Droffelaar et al., 2022).

Table 2.4: Effect of search space reduction on the possible values for the decision variables (DVs) and the number of permutations, averaged over 50 seeds. Without applying the search space reduction techniques, the number of permutations is $|V|^{|U|}$, where $|V|$ is the number of vertices in the network and $|U|$ is the number of police units.

Problem size		Avg. reduction per DV	Nr. of permutations (% of unfiltered)
$\|U\|$	$\|V\|$		
1	900	86.3%	1.23e2 (13.7%)
5	900	81.1%	1.52e11 (0.026%)
10	900	85.6%	9.37e20 (2.69e-7%)
5	100	36.5%	1.08e9 (10.8%)
5	2500	90.0%	2.56e12 (0.0026%)

In each experiment, one fugitive and $|U|$ police units are placed on random vertices. We sample 500 fugitive routes. The fugitive is intercepted if it occupies the same vertex at the same time as a police unit, as defined in equations 2.4 and 2.7. We optimize the target vertex of each of the police units to maximize the number of intercepted fugitive routes. We evaluate the performance for 10 different combinations of starting locations of police units and the fugitive.

Table 2.5 details the parameter ranges used in the experiments. The number of vertices in the network and the number of police units to be positioned are chosen to be realistic for the application. The network size determines the length of the planning horizon, meaning the maximum time in the model. It is chosen so that it is possible to just traverse the network within L minutes. Specifically, L is 5 + (0.5 $*\sqrt{|V|}$) for the 2D grid and the radius of the road network in meters, divided by 5 for the city road network. The number of fugitive routes simulated in the model is chosen to be sufficient to cover the network. We optimize 10 different combinations of starting locations of police units and the fugitive to account for the varying complexity between combinations of starting locations. Some instances of the problem may be much easier to solve due to a convenient starting location. By sampling 10 different combinations, we control for this variance. To control for the stochastic processes in the optimization algorithm, we run each experiment for 5 seeds. All experiments are conducted on the same machine, the DelftBlue supercomputer (Delft High Performance Computing Centre, 2022) on a dedicated node to prevent interference. This cluster offers an Intel XEON E5-6248R 24C 3.0GHz CPU with 48 cores and 192 GB memory.

2.3.5. ROAD NETWORKS

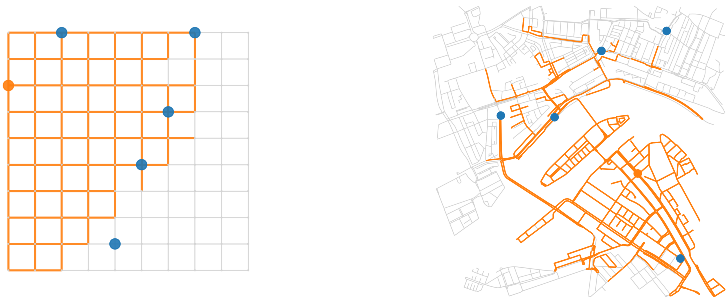
The topology of the road network dictates the patterns in the fugitive routes and the relative reachability of parts of the network. To account for the effect of the topology, the experiments are performed on two networks: a 2D 'Manhattan' grid with an edge travel time of 1 minute and an extract of the road network of Rotterdam, a typical European city. A grid network is suggested by Rydzewski and Czarnul (2020) to improve cross-study comparison of methods and algorithms. The

Table 2.5: Parameter ranges for the experiments. The length of the planning horizon (L) depends on the network type.

Parameter	Range	Default value
Number of units to be positioned ($\|U\|$)	1-10	5
Number of vertices in network ($\|V\|$)	100-2500	900
Length of planning horizon (L)	n.a.	$5 + (0.5 * \sqrt{\|V\|})$; radius (m) / 5
Number of routes ($\|R\|$)	n.a.	500
Number of starting positions	10	n.a.
Number of random seeds for metaheuristic	5	n.a.

city road network is extracted from OpenStreetMap[4] using Boeing's Python library OSMnx (Boeing, 2017). We vary the radius (in meters) from a central point in the city, resulting in varying-sized networks. For the 2D grid, we vary the diameter, yielding networks of varying sizes.

Examples of each network and a combination of starting locations are presented in Figure 2.2.



(a) A 100-vertices Manhattan grid

(b) A 689-vertices Rotterdam road network

Figure 2.2: Test networks, where the starting position and sampled routes of the fugitive are indicated in orange, and the starting positions of the police units are indicated in blue.

We use the best possible solution obtained by the exact optimization algorithm to assess the convergence of the metaheuristic optimization algorithm for the equidistant grid case. The MIP optimization problem is indexed on time, which works nicely for an equidistant graph such as the 2D grid in the first set of experiments. Since the vertices in a real-world road network are not equidistant, time discretization has implications for the accuracy of the optimization model. For a city road network, a very small time step of 1 second is needed to avoid discretization errors and obtain accurate results to assess the quality of the metaheuristic. This leads to high computation times that exceed the time constraint for real-time decision making. Therefore, the best-found solution across seeds is used as the reference set for the city road network.

[4]OpenStreetMap: https://www.openstreetmap.org

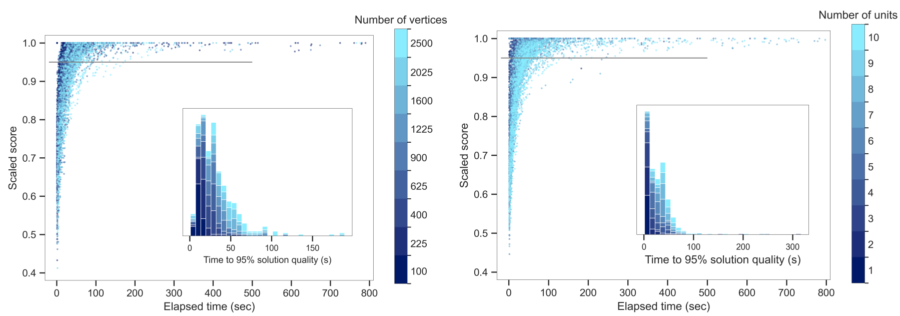
2.4. RESULTS AND DISCUSSION

2.4.1. CASUS 1: GRID TEST GRAPH

We perform the first set of experiments on a 2D 'Manhattan-like grid. The following paragraphs describe the results for simulation model optimization and sequential simulation-optimization individually, followed by a comparison of the approaches.

SIMULATION MODEL OPTIMIZATION

We examine the effect of increasing problem size on the convergence of the Borg algorithm using the simulation model optimization configuration. We see that the convergence speed decreases with increasing problem size (Figure 2.3). With an increasing number of police units and number of vertices, the density of the improvements shifts downwards (to lower quality of results) and to the right (to longer computation time). The inset histograms demonstrate that the time to 95% of the maximum attainable solution quality increases with increasing problem complexity. The majority of optimization instances reach this quality within 100 seconds, with outliers for large networks up to 200 seconds and outliers for a large number of police units up to 320 seconds.



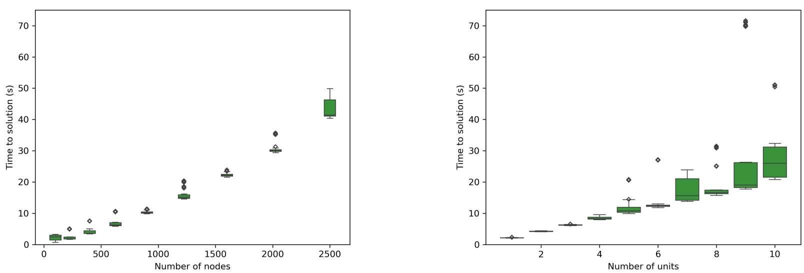
(a) Varied number of vertices

(b) Varied number of police units

Figure 2.3: Convergence of the metaheuristic on simulation model optimization for varying problem size. Each dot indicates an improvement found by the algorithm. The insets portray a histogram of the time to 95% solution quality.

SEQUENTIAL SIMULATION-OPTIMIZATION

Exact solution approach Using the exact algorithm COIN-OR Branch-and-Cut (CBC), the time to solution increases with increasing problem size (Figure 2.4). Given a fixed network size of 900 vertices, the computation time appears to increase linearly with an increasing number of police units (Figure 2.4b). This is because the model treats each police unit independently: adding another police unit adds another decision variable with the same number of options. The interaction effects between police unit interceptions are not explicitly modeled but rather are reflected by the expected number of intercepted routes. With an increasing network size, the computation time increases faster than linearly (Figure 2.4a). This can be explained by the number of possible combinations between police units and target vertices increasing factorially. With 5 police units, the time to find the optimal solution increases from less than a minute for small network instances with 100 vertices to over 13 minutes for larger network instances with 2500 vertices. A typical city road network in the Netherlands consists of 2000-4000 vertices. A computation time of over 10 minutes is unacceptable for real-world application. Moreover, the variance of the computation time increases with increasing problem size, though the predictability of the computation time is crucial for real-time decision-support.



(a) Varied number of vertices

(b) Varied number of police units

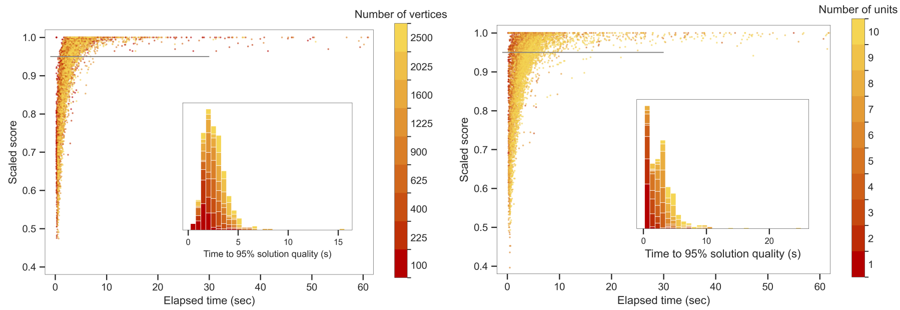
Figure 2.4: Computation time of exact solver CBC on sequential simulation–optimization for varying problem size.

Metaheuristic solution approach We examine the effect of increasing problem size on the convergence of the metaheuristic algorithm using the sequential simulation-optimization configuration. The solution quality is scaled to the optimal solution found by the exact solver CBC.

Using the metaheuristic algorithm Borg, we see that the convergence speed decreases with increasing problem instance size (Figure 2.5). With an increasing number of police units, the density of the improvements shifts downwards (to lower quality of results) and to the right (to longer computation time). This effect is also visible, though to a lesser extent, with an increasing number of vertices. The inset histograms demonstrate that the time to 95% of the maximum attainable solution quality increases with increasing problem complexity. The majority of cases reach this quality within 10 seconds, with outliers for large networks up to 16 seconds and outliers for a large number of police units up to 25 seconds.

COMPARISON

To compare the computation time of the two configurations and two solution approaches, we calculate the elapsed time at which each problem instance reached a scaled score of at least 0.95. The scaled score is the quality of each solution divided by the best-found solution for the problem instance by the exact MIP solver.



(a) Varied number of vertices

(b) Varied number of police units

Figure 2.5: Convergence of the metaheuristic on sequential simulation–optimization for varying problem size. Each dot indicates an improvement found by the algorithm. The insets portray a histogram of the time to 95% solution quality.

Figures 2.6a and 2.6b show that sequential simulation optimization, especially when solved using a metaheuristic approach, outperforms simulation model optimization. The computation time of simulation model optimization is spread out: for example, for $|V| = 2500$, the time to 95% solution quality varies from 15 seconds to 140 seconds. For application in a real-world control room, a near ten-fold variance in the time to solution is unacceptable.

The difference in (variance of) computation time between simulation model optimization and sequential simulation-optimization, both solved using a metaheuristic, is mainly caused by the different computation time per function evaluation. The number of function evaluations to convergence is similar for both simulation-optimization configurations. Each function evaluation, a full-fletched simulation model containing both fugitive and police entities is run to completion. This is obviously more time-consuming than sequential simulation- optimization, where, after preprocessing, each function evaluation involves comparing matrices to count interceptions.

Some problem instances with $|U| = 9$ or $|U| = 10$ pose difficulty for the CBC MIP solver. Most likely, the corresponding starting configurations do not allow for easy exploitation of the problem structure by CBC and force the algorithm to evaluate many candidate solutions. These problem instances do not cause a similar effect on the convergence speed of the metaheuristic.

2.4.2. CASUS 2: REAL-WORLD ROAD NETWORK

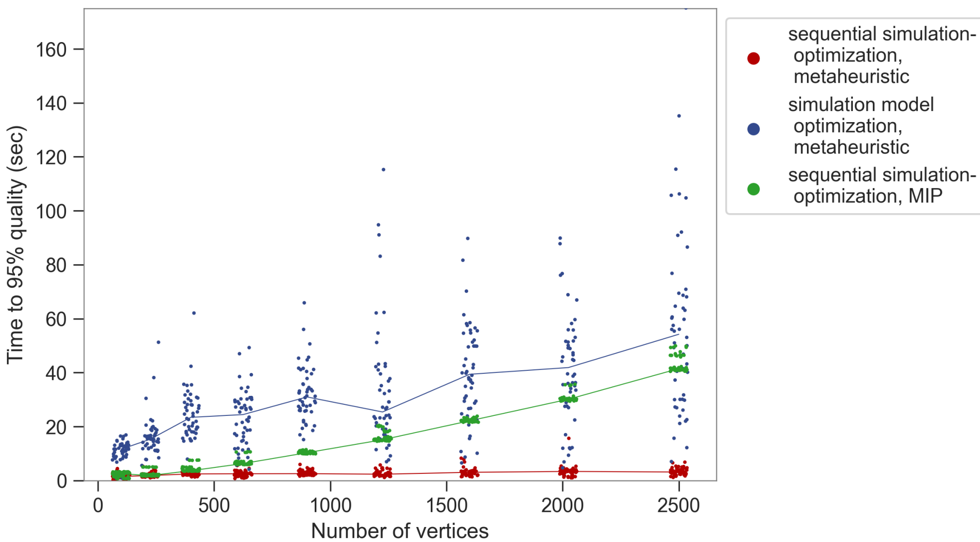
The same set of experiments on the road network of Rotterdam, the Netherlands, demonstrate the generalizability of the results to different graph topologies. Due to the discretization error in the optimization on city road networks, we do not consider the exact optimization approach. Figure 2.7 shows that the computation time of simulation model optimization further increases compared to the experiments on a simple grid due to additional computational overhead. The reliability of both solution approaches is lower due to the larger influence of the starting positions of the fugitive and police units on the number of feasible permutations.

With an increasing number of police units to be positioned, the median computation time increases gradually until $|U| = 7$, after which it decreases. Evidently, it is relatively easy to find high-quality interception strategies using many police units on a real-world road network compared to a grid. There are relatively fewer good interception positions, which decreases the solution space, leading to faster convergence.

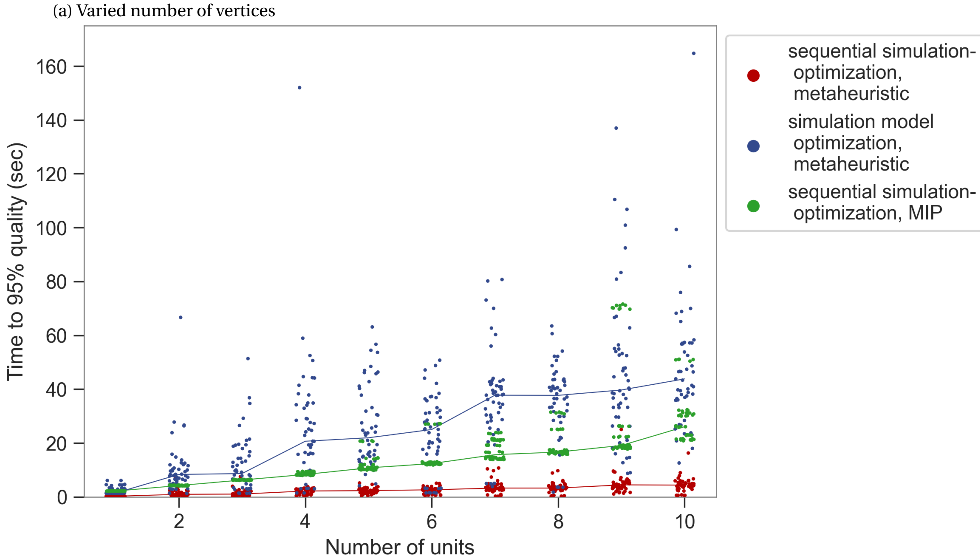
2.4.3. DISCUSSION

The experiments show that sequential simulation-optimization significantly decreases the computation time compared to simulation model optimization of the same problem. Sequential simulation-optimization lends itself well to problems where the external, uncontrollable factors can be separated from the controllable factors. Simulation-optimization is often approached as the 'bolting on' of an optimization engine on an existing simulation model (Henderson, 2021). This approach is not easily transformed into a sequential simulation-optimization model. Firstly, the problem at hand should be suitable for implementation in the particular configuration. Modeling complexity in the controllable factors is more difficult compared to a simulation model optimization approach, as this complexity has to be described by constraints. In the considered fugitive interception problem, the external factor is the fugitive's behavior, and the controllable factor is the behavior of the police units. Hence, introducing more complex behavior of the police units is more difficult when using sequential simulation-optimization compared to simulation model optimization. Second, sequential simulation-optimization requires a specific formulation of the simulation model that yields the constraints for the optimization. This model formulation differs from the typical simulation model used to answer 'what-if'-type questions about the system under study.

The interception problem described in this paper is an example of a class of problems where an optimal intervention has to be determined independent of the uncertainty in the system. Therefore, controllable and uncontrollable components are separable into optimization and simulation, respectively. Another example of this class of problems is the control of an autonomous vehicle. The best control action has to be determined, meaning that the car follows its intended route and avoids crashes, while the behavior of the vehicles around it is unknown. Similar to the fugitive interception problem discussed in this paper, the control action must be available quickly - within a second or even less. To accomplish this using sequential simulation-optimization, the possible trajectories of the nearby road users are simulated. A robust optimization yields the control action (Zanon et al., 2014). Sequential simulation-optimization is a promising way to quickly generate a large ensemble of plausible routes, incorporate these in an optimization problem, and find the optimal control action for the autonomous vehicle.

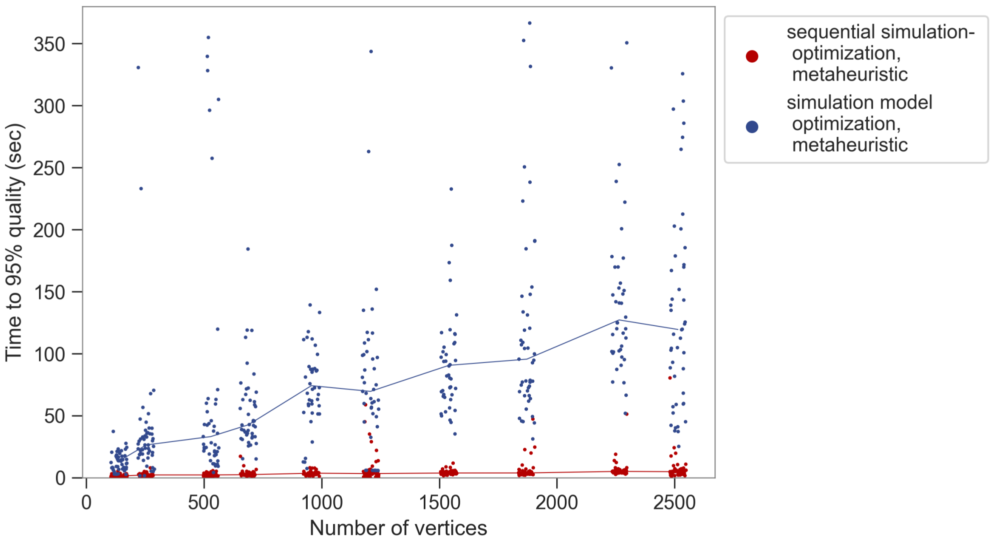


(a) Varied number of vertices

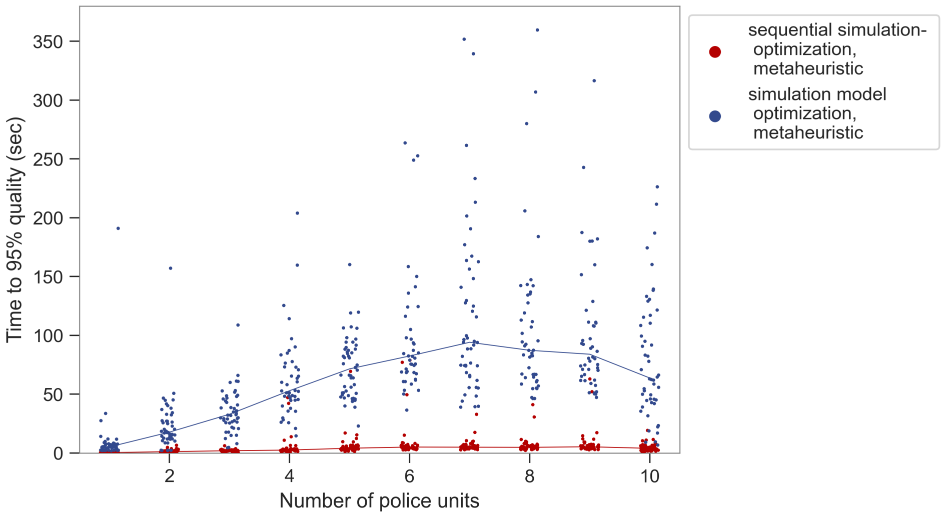


(b) Varied number of police units

Figure 2.6: Comparison of the computation time for a varying problem size for three approaches: simulation model optimization (solved using a metaheuristic) and sequential simulation–optimization (solved with an exact optimization algorithm and a metaheuristic). The lines indicate the median time to 95% solution quality for each approach.



(a) Varied number of vertices



(b) Varied number of police units

Figure 2.7: Comparison of the computation time for a varying problem size for simulation model optimization and sequential simulation–optimization. The lines indicate the median time to 95% solution quality for each approach.

2.5. THREATS TO VALIDITY

The first threat to validity concerns the experimental setup. The computational experiments are based on a default configuration with 900 vertices (a 30x30 square grid or an 8km^2 patch of the Rotterdam road network) and 500 predicted escape routes for the fugitive. In a typical fugitive route-choice simulation, approximately 500 routes are adequate to describe the plausible escape routes of the fugitives if only considering highways. If minor roads are included, the additional intersections cause this number to increase. A network size of approximately 1000 vertices is required to adequately describe the main network around a typical urban incident location. If the network is extended to minor roads, this number increases rapidly, analogous to the number of predicted escape routes required. In practice, the road network grows with the desired length of the planning horizon. The number of vertices that describe the road network can be reduced through graph coarsening (Krishnakumari et al., 2020). However, the extent to which graph coarsening can be applied is limited: interception loses its real-world meaning if the network used in the model is too coarse. Therefore, the rapid increase in computation time with increasing vertices is troublesome, as the timely calculation of relevant solutions is threatened.

Secondly, the tested approaches may be more or less suitable for further reduction of the computation time. There are two main approaches to reducing the computation time of both simulation-optimization configurations: (1) increasing the computation power, for example, through further parallelization with High-Performance Computing; (2) improving the optimization algorithm. The former reduces the computation time per function evaluation. Increasing the computation power improves the absolute computation time, but the computation time scales the same way with increasing problem size as presented in this paper. The same holds for replacing Python with a computationally more efficient language, such as C. Furthermore, the improvement from parallelization is dependent on the specific optimization algorithm used. For many algorithms, for example, those based on hill-climbing, the optimization algorithm depends on the simulation model's output for its next proposed set of values for the decision variables. In this case, the improvement from parallelization is limited. In this paper, the applied algorithm, the input space, and the solution space are the same for both compared simulation-optimization approaches. Therefore, each of the discussed approaches to further reduce the computation time is expected to affect the computation time of both approaches similarly.

2.6. CONCLUSION

Simulation-optimization can be used to support real-time decision making for fugitive interception. To be useful for real-time decision making, timely calculation of the optimal solution is essential. Besides increasing computation power and algorithm efficiency, the configuration in which simulation and optimization are combined can reduce the computation time of simulation-optimization of large problems.

This paper examined the scaling of computation time with increasing problem size for two configurations of simulation-optimization: (1) sequential simulation-optimization: the output of a simulation model describes (part of) the constraints of an optimization model; (2) (common) simulation model optimization: a simulation model evaluates values for the decision variables proposed by an optimization algorithm to find the values that maximize or minimize the outcome(s) of interest, which are calculated through the simulation model. Our analysis using the fugitive interception example showed that

1. Sequential simulation-optimization vastly outperforms simulation model optimization in terms of computation time, especially for large problem instances.
2. Metaheuristic solution approaches reach a high quality of solutions in a fraction of the computation time of exact optimization algorithms.
3. Experiments on a real-world city road network demonstrate that these findings hold for various graph topologies.

These results are generalizable to a class of problems where an optimal intervention has to be determined independent of the uncertainty in the system. In other words, separating controllable and uncontrollable components into optimization and simulation, respectively, leads to a significant reduction in the computation time.

3

GRAPH COARSENING

The timeliness of the calculated police interception positions has been a challenge in the previous chapter, especially when testing approaches on full-scale city road networks. Therefore, this chapter proposes a graph coarsening approach for fugitive interception to improve the timeliness without compromising the quality of the police interception.

This chapter is published as: Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2025). Graph coarsening for fugitive interception. *Applied Network Science, 10,* 2.

The code and data associated with this chapter are available at: doi:10.4121/07643762-6038-4ccc-bf94-4bf56b5abeae

ABSTRACT

The police control room determines where to send available police units to intercept a fleeing fugitive. Models can support the police with decision-making for fugitive interception. The police have, at most, a few minutes to determine an interception strategy. Therefore, a timely calculation of the interception positions is essential to support police interception operations. The number of nodes in the network, each being a crossing where routes of the fleeing suspect can split, greatly contributes to the computation time. Graph coarsening is a promising approach to reduce the complexity of the network, and therefore the computation time. We compare four graph coarsening algorithms on five road networks and assess their impact on computation time and solution quality for the fugitive interception problem. Based on the comparison, we propose and test a new method specifically for fugitive interception. This method, Search Space Representation, improves the quality of the best solutions obtained by the optimization algorithm by up to 12%, improves the reliability of the optimization to find high-quality solutions, and decreases the number of function evaluations required to obtain high-quality solutions to 5 000 - 10 000 depending on the size and complexity of the road network, which is feasible for real-time decision-making. Search Space Representation can be applied to reduce the computation time of other network-based optimization problems.

3

3.1. INTRODUCTION

Fugitive interception is a challenging task, requiring police to decide in at most a few minutes on the optimal positions of police units to intercept a fleeing suspect. The fleeing fugitive moves from the incident location to escape interception, e.g., by crossing the border or reaching the highway. Police units do not know the fugitive's whereabouts, so they have to move to a vertex in the network where the probability of interception is highest, e.g., a chokepoint in the network where many routes pass through. The goal of the so-called 'fugitive interception problem' is to position the police units in such a way that they maximize the number of intercepted escape routes. Possible escape routes are simulated using a generative model of escape behavior (van Droffelaar et al., 2024a). Note that the fugitive interception problem is not about chasing a fleeing fugitive; it concerns *intercepting* a fleeing fugitive, who is taking an unknown escape route from a known crime location. Models can support police decision-making for this problem, but the timely calculation of optimal interception positions is challenging due to the complexity of the problem. A major contributor to the overall computation time is the size of the underlying road network, particularly the number of nodes in the network (van Droffelaar et al., 2024b).

Graph coarsening, a technique to reduce the size of a graph while preserving essential structural properties, offers a promising approach to reducing the computation time of the fugitive interception problem. Graph coarsening is also referred to as contraction hierarchies (Geisberger et al., 2008), graph reduction (Loukas, 2019), edge contraction (Asano & Hirata, 1983), and graph sparsification (Peleg & Schäffer, 1989). Graph coarsening algorithms have successfully been applied to various graph-based optimization problems where reducing the number of nodes significantly improves computation time, such as routing optimization (Sanders & Schultes, 2012), the Traveling Salesman Problem (Walshaw, 2004) and graph partitioning (Chevalier & Safro, 2009).

Graph coarsening algorithms vary depending on the application, as the importance of the nodes and links is very case-specific. For example, coarsening for transport modeling often focuses on preserving the shortest paths (Sanders & Schultes, 2012), while coarsening for graph partitioning aims to minimize the number of edges (Chevalier & Safro, 2009; Safro et al., 2015). Pung et al. (2022) propose an algorithm that coarsens road networks using characteristics most prominent in the United States – grids and cul-de-sacs.

We distinguish two approaches to graph coarsening with different trade-offs between solution quality and computation time for fugitive interception: pre-processing and on-the-fly coarsening. The first approach coarsens networks in advance and saves them for later application. Hence, the computation time of the coarsening algorithm does not affect the computation time of any subsequent optimizations. However, the coarsening algorithms cannot take any incident-specific information into account, like the starting positions of police units, the starting position of the fugitive, or their plausible escape routes. Therefore, pre-processed coarsening risks removing nodes of high importance to any specific fugitive interception problem and, therefore, decreasing the solution quality. In contrast, on-the-fly coarsening approaches can take all relevant incident characteristics into account, likely leading to a higher solution quality. However, the computation time of the coarsening is critical in this case and might offset any gains in computation time for the optimization.

This paper compares four graph coarsening techniques on both computation time and solution quality for fugitive interception. We measure the solution quality by running the optimization algorithm for 100 000 function evaluations across 10 seeds and take the best-found solution. To measure the computation time, we consider (1) the number of function evaluations (NFE) required by the optimization algorithm to find a solution, and (2) the computation time per function evaluation. We evaluate these methods across five different types of road networks. The evaluation studies both pre-processing and incorporating on-the-fly graph reconstruction into the optimization process.

The contribution of this research is two-fold: (1) we compare the effectiveness of existing graph coarsening algorithms for a new application, and (2) we propose an approach incorporating on-the-fly graph reconstruction into the Search Space Representation in the optimization process. This allows for more flexibility, capable of handling different fugitive profiles and network structures.

Section 3.2 describes the case study used throughout the paper, fugitive interception optimization. Section 3.3 describes the experimental setup. Section 3.4 details each of the chosen algorithms, their implementation for fugitive interception, and the results of the experiments. Based on the results, Section 3.5 proposes and tests an on-the-fly coarsening approach for fugitive interception. Section 3.6 discusses the possible threats to the validity and implications of the research, and, lastly, we share our conclusions in Section 3.7.

3.2. OPTIMIZATION PROBLEM

3.2.1. BACKGROUND

The fugitive interception problem aims to find the best positions for police units to maximize the probability of intercepting a fleeing fugitive on a road network. The problem is modeled from the start of the escape from the incident until the fugitive is either intercepted or escaped. Since the police do not know the fugitive's exact location, they have to position themselves at points in the network where there is a high chance of interception, such as chokepoints where many escape routes intersect.

Related optimization problems in literature are search, and, more specifically, interception problems. Alspach (2004) optimizes the routes of searchers to either maximize the probability or minimize the time to find a target. However, determining an optimal action for each time step becomes computationally infeasible when the size of the network and the length of the time horizon increase.

Pursuit-evasion games, where both the routing of the searcher (or pursuer) and the target (or evader) are optimized, are primarily used in robotics (Chung et al., 2011). Pursuit-evasion problems are solved for different graph topologies, such as grids, circular graphs, trees, and random graphs. Depending on the network topology, these problems are proven to be pseudo-P to strongly NP-complete (Borie et al., 2011). Due to the computational complexity, the problem instances that are studied are typically very small.

3.2.2. FORMALIZATION OF THE OPTIMIZATION PROBLEM

We formulate the optimization problem as a variant of the Flow Interception Problem (Berman et al., 1992; Hodgson, 1990), meaning the objective is to position each police unit to maximize the number of intercepted escape routes. A route is considered intercepted if (1) it passes a police unit's target position and (2) the police unit can reach that position before the fugitive does.

The decision variables of the optimization problem are the target nodes of the police units ($\pi_{u,v}$) and the intercepted routes (z_r) (Table 3.1). The optimization problem is formalized in Equations 3.1-3.3. A route is considered intercepted ($z_r = 1$) if, for a given route (r), the fugitive is at the same place (v) at the same time (t) as the position of a police unit ($\pi_{u,v}$), and that position is within reach at that time for that particular police unit ($\tau_{u,v,t}$). The positions of the police units are optimized to maximize the number of intercepted escape routes. Routes are only intercepted at target positions, not at intermediate points along the route. The minimization

Table 3.1: Notation of parameters and decision variables.

Decision variables	
$z_r \in \{0,1\}$	route r is intercepted
$\pi_{u,v} \in \{0,1\}$	node v is the target node of police unit u
Parameters	
$V = \{v\}$	set of nodes
$R = \{r\}$	set of fugitive routes
$U = \{u\}$	set of police units
$S = \{s\}$	set of sensors
$T = \{t\}$	ordered index set of time steps
t_{max}	maximum time step; length of planning horizon
$\phi_{r,v,t} = \{0,1\}$	fugitive route r is present at node v at time step t
$\tau_{u,v,t} = \{0,1\}$	node v is reachable by police unit u at time t

function in Equation 3.3 ensures that each route can only be intercepted once and contribute to the objective function.

$$\text{Maximize: } Z = \sum_{r \in R} z_r \tag{3.1}$$

$$\text{Subject to: } \sum_{v \in V} \pi_{u,v} = 1 \qquad \forall u \in U \tag{3.2}$$

$$z_r = \min\left(1, \sum_{u \in U} \sum_{t \in T} \sum_{v \in V} \phi_{r,v,t} \cdot \pi_{u,v} \cdot \tau_{u,v,t}\right) \qquad \forall r \in R \tag{3.3}$$

3.2.3. SIMULATION OF THE FUGITIVE ESCAPE ROUTES

The optimization depends on the simulated escape routes of the fugitive. The routes are modeled as the shortest paths from the incident location to the escape nodes of the network. To generate a diverse set of plausible routes, 2% noise is added to the routes, meaning that the fugitive makes a wrong turn at 2% of the intersections, after which a new shortest path is recalculated from their current position (van Droffelaar et al., 2024a). This approach produces a distribution of routes around the optimal paths. Simulating the fugitive escape routes through this method takes a few seconds, depending on the road network, the starting position of the fugitive, and the locations of the escape nodes (Winterswijk: 2.4 s, Utrecht: 9.9 s, Manhattan: 1.4 s, Main roads: 1.1 s, Rotterdam: 9.2 s). The simulation of fugitive routes should be further optimized and parallelized before implementation in a control room. In future work, this model could be replaced with a more detailed behavioral model.

3.2.4. SOLUTION APPROACH

The optimization problem is NP-hard, meaning that exactly solving real-world cases could take years (Boccia et al., 2009). Therefore, we use a genetic algorithm supplemented with the auto-adaptive framework from Borg, which co-evolves the probabilities of the evolutionary operators used for population adaptation based on their success in generating better solutions (Hadka & Reed, 2013). van Droffelaar et al. (2024b) compare this metaheuristic optimization algorithm to the exact optimization algorithm CBC (Ralphs, 2022) and show that the metaheuristic finds near-optimal solutions in a fraction of the computation time. Problem instances on networks with 2 500 nodes are solved in 5-10% of the computation time, and the computation time increases less rapidly with increasing network size. To speed up convergence, the nodes are sorted on their proximity to the starting position of the fugitive. Thus, the proximity of solutions in the search space is more related to proximity in the objective space. To further ensure solution quality, the optimization algorithm is run for ten random seeds for 100 000 function evaluations. All experiments are performed on the Delft Blue supercomputer, with an Intel XEON E5-6248R 24C 3.0 GHz CPU with 48 cores and 192 GB memory (Delft High Performance Computing Centre, 2022).

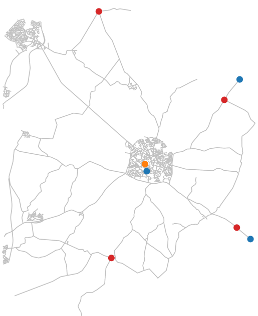
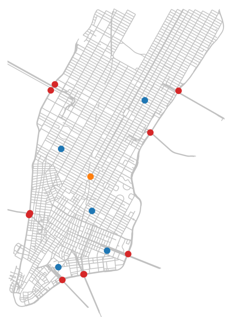
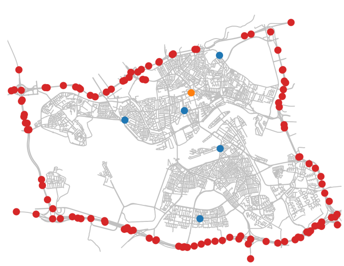
3.3. METHOD

3.3.1. CASE STUDY NETWORKS

To reduce the dependency of the experiments on the topology of the road network, we evaluate the coarsening approaches on five distinct road networks (see Table 3.2 for an overview). First, Winterswijk, the Netherlands, represents a rural area. Sparse roads lead from the town to the border with Germany in the north, east, and south. Second, Manhattan, New York, United States of America, represents a grid layout city with traffic lights and cameras at most intersections. Third, Utrecht, the Netherlands, represents a typical European city with a historical center surrounded by more modern neighborhoods. Fourth, the main road network around Amsterdam, the Netherlands, consists of highways, and primary and secondary roads around the city. This network is vastly different from city road networks and represents an application where a fugitive flees at high speeds over a larger distance. Fifth, Rotterdam, the Netherlands, represents a large modern European city divided by a large river. The starting position of the fugitive is a central location in each road network, and the police start locations are the local police stations in the respective areas.

The networks are obtained from OpenStreetMap via the OSMnx Python library (Boeing, 2017; Boeing, 2024). We use the built-in `simplify_graph` functionality to remove nodes that do not represent intersections, as well as dead ends. The OpenStreetMap raw data consists of sets of straight-line segments: curved roads contain intermediate nodes to represent their geometry. For fugitive interception, these nodes do not add any value but do increase the number of nodes considerably. For example, the unsimplified Winterswijk network consists of 9540 nodes, whereas simplifying the network reduces the number of nodes to 1926.

Table 3.2: Case study road networks used in this study. Escape nodes are marked in red. The starting positions are displayed in blue (police units) and orange (fugitive).

Winterswijk	Manhattan	Utrecht
Rural area Incident: city center Escape: cross the border Size: 1926 nodes	Modern, grid city Incident: Union Square Escape: get off the peninsula Size: 2533 nodes	Typical European city Incident: city center Escape: reach the highway Size: 4557 nodes
		
Main road network	**Rotterdam**	
Typical highway network Incident: Amsterdam Escape: network edges Size: 3241 nodes	Modern European city Incident: city center Escape: reach the highway Size: 7108 nodes	
		

3.3.2. EVALUATION METHOD

Each coarsening algorithm is evaluated on each of the road networks using the framework depicted in Figure 3.1. The evaluation method consists of four steps:

1. The fugitive escape routes are simulated on the uncoarsened road network G using the method described in Section 3.2.3.
2. The road network G is coarsened using the algorithm under evaluation, resulting in the coarsened network G_c. Each coarsening algorithm has its own tuning parameters, which are varied to obtain different extents of coarsening for each algorithm.
3. The police interception positions are optimized based on the coarsened network G_c, meaning that only nodes that remain in G_c are considered possible interception positions. The number of routes intercepted by a combination of interception positions – the quality of the candidate solution – is calculated using the set of routes generated in step 1.
4. The police interception positions optimized based on G_c are then evaluated on the original graph G. Discrepancies in the number of intercepted escape routes arise when either the path between the police start position and the calculated interception position is longer in G than in G_c, or when the path does not exist in G_c or in G. We collect three metrics from each parametrization of each coarsening algorithm:
   - The *solution quality*, which is the percentage of escape routes intercepted by the calculated police interception positions. Note that the metaheuristic solution approach does not guarantee finding the exact optimal solution. Therefore, we take the best-found solution after 100 000 function evaluations across 10 seeds to account for variations in convergence between seeds. The solution quality is scaled to the best-found solution quality on the uncoarsened graph G to obtain the degradation of the solution quality caused by the coarsening of the graph.
   - The *convergence*, which is the number of function evaluations at which the optimization algorithm obtains 95% of its solution quality. Effectively, this is the number of function evaluations at which the search stalls. To account for variation between seeds, we take the minimum number of function evaluations at which a seed reaches 95% of its best-found solution quality. This statistic is collected for every seed of each parametrization of each coarsening algorithm.
   - The *time per function evaluation*, which, combined with convergence, indicates how much graph coarsening reduces the computation time. For these experiments, we rerun a subset of the coarsening algorithm parametrizations on a dedicated node of the supercomputer to prevent interference with other jobs.

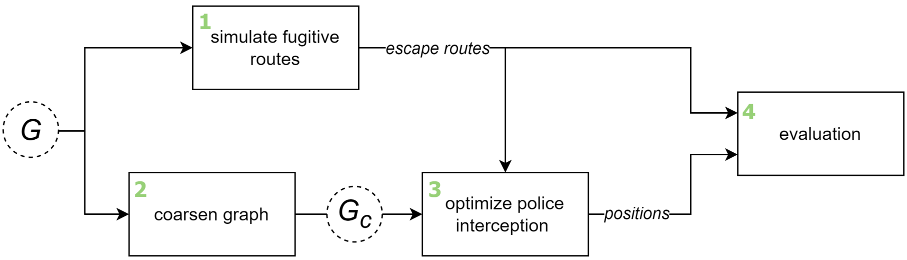


Figure 3.1: Schematic representation of the evaluation method used in this study.

3.4. COARSENING ALGORITHMS: RESULTS AND DISCUSSION

We compare the effectiveness of four graph coarsening algorithms for fugitive interception: three preprocessing algorithms and one on-the-fly approach. The three preprocessed graph coarsening algorithms are selected for their diverse approaches to graph coarsening and the availability of open-source implementations (preferably in Python). The on-the-fly coarsening method was developed as part of this research. The algorithms considered are:

1. *Pruning*: a first step to simplifying the network, by removing dead ends and cul-de-sacs (Pung et al., 2022).

2. *Node consolidation*: a generic, application-agnostic graph coarsening approach that merges nodes that are topologically close together to simplify complex intersections and clusters in the network (Boeing, 2024).

3. *Heuristic coarsening*: a transport-specific coarsening algorithm that preserves key properties of the network, such as connectivity and shortest paths (Krishnakumari et al., 2020).

4. *On-the-fly coarsening*: a case-specific coarsening algorithm that filters the network before optimization, retaining only the nodes and edges that are relevant to the specific interception case.

The following paragraphs outline the four graph coarsening algorithms, first discussing them in general, second detailing the implementation for the fugitive interception case used throughout this paper, and third, presenting the results. Lastly, the computation time and obtained solution quality of the algorithms are compared.

3.4.1. PRUNING

BACKGROUND

A logical first step to simplify the road network is pruning dead ends and cul-de-sacs (Pung et al., 2022). Dead ends and cul-de-sacs are not expected to be nodes with a high probability of intercepting a fleeing suspect. Therefore, pruning these nodes should not degrade the solution quality by much.

IMPLEMENTATION

This study uses a simple recursive algorithm that removes nodes with only one incoming or outgoing edge. Self-loops, which are edges that connect a node to itself, are also removed. The recursion ensures that any new dead ends or self-loops created during the pruning process are also removed.

RESULTS AND DISCUSSION

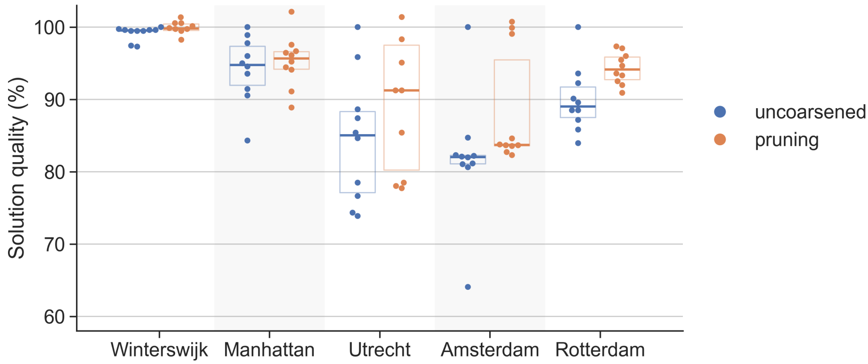
Depending on the road network, the number of nodes in the network is reduced by 2.7% to 29.1%, depending on the road network (Table 3.3). This is a considerable reduction, especially because the removal of these nodes does not affect the solution quality. Figure 3.2a shows that, relative to the uncoarsened graph, pruning obtains the same best-found solution quality across seeds. Notably, the mean obtained solution quality increases for all road networks. Note that for some seeds, the solution quality exceeds 100%, indicating that the metaheuristic solution approach identifies solutions better than the best-known solutions for the uncoarsened network. This occurs because metaheuristics do not guarantee finding the exact optimal solution. The reduction in the number of nodes available for interception decreases the size of the solution space, which speeds up convergence and reduces the likelihood of getting stuck in local optima.

Additionally, the variation between seeds either stays similar or decreases. This is important for the predictability of computation time for the decision-maker. A large variation in solution quality across seeds indicates that a decision-maker should run with many seeds in parallel and aggregate the results.

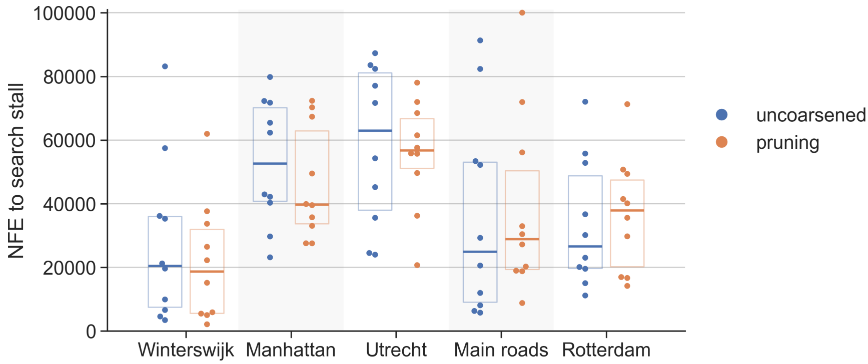
Figure 3.2b shows that the convergence of the optimization algorithm is largely dependent on the seed and the initial sample of solutions. On average, pruning leads to slower convergence in most road networks. This, however, is a distortion of the plot due to the much higher obtained solution quality (Figure 3.2a). The earliest NFE at which the optimization algorithm finds a solution with the quality of the uncoarsened network solution is lower after pruning.

Table 3.3: Node reduction by pruning for the five case study road networks, relative to the uncoarsened networks.

City	Node reduction (%)
Winterswijk	29.1
Manhattan	2.7
Utrecht	15.1
Main roads	5.9
Rotterdam	13.0



(a) Impact of pruning on the solution quality. The results are scaled to the best-found solution quality across seeds for the uncoarsened network.



(b) Impact of pruning on the convergence. The NFE to search stall is the NFE at which that seed reached 95% of its best-found solution quality.

Figure 3.2: Results of the pruning experiments with ten seeds per city. Each dot represents the results of one random seed. The overlaid boxes represent the first quartile, median, and third quartile.

3.4.2. NODE CONSOLIDATION

BACKGROUND

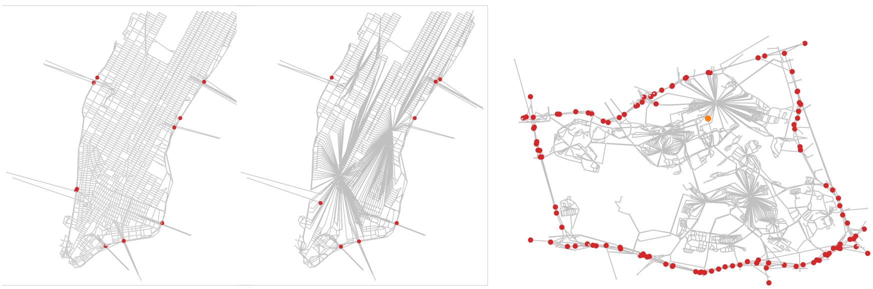
Node consolidation is a graph simplification method that merges nodes that are topologically close together (Boeing, 2024). Real-world road networks often have complex intersections that, when converted to a graph, result in a group of nodes. For instance, a roundabout is represented by four or eight nodes, depending on its specific layout. For transport planning – and for fugitive interception – we can consider these multiple nodes as one.

IMPLEMENTATION

This study uses an OSMnx's node consolidation algorithm (Boeing, 2024). The algorithm's tuning parameter, 'tolerance' (measured in meters), defines the buffer radius around each node. Overlapping node buffers are merged into a single node at the center of the buffer area. In this study, we vary the tolerance from 1 to 50 meters. A low tolerance simplifies complex intersections into single nodes (Figure

3.3a). A higher threshold collapses more of the network, but keeps the main roads intact (Figure 3.3c). A grid network, like Manhattan, collapses after surpassing a tolerance threshold of the distance between the blocks (Figure 3.3b).

The node consolidation algorithm retains information about which original nodes were consolidated into each new node. After consolidation, the starting positions of the police, the fugitive, and the escape nodes are mapped to the corresponding consolidated nodes. By handling this in post-processing, rather than exempting certain nodes during consolidation, the coarsened network is flexible to any incident location.



(a) Manhattan (10m). (b) Manhattan (45m). (c) Rotterdam (50m).

Figure 3.3: Road networks after node consolidation with different tolerance settings (in brackets).

RESULTS AND DISCUSSION

Table 3.4 show that varying the 'tolerance' parameter produces coarsened networks with varying numbers of nodes. The exact impact of tolerance differs across networks. When the tolerance is set to its maximum of 50 meters, some networks are reduced to as little as 7% of their original size.

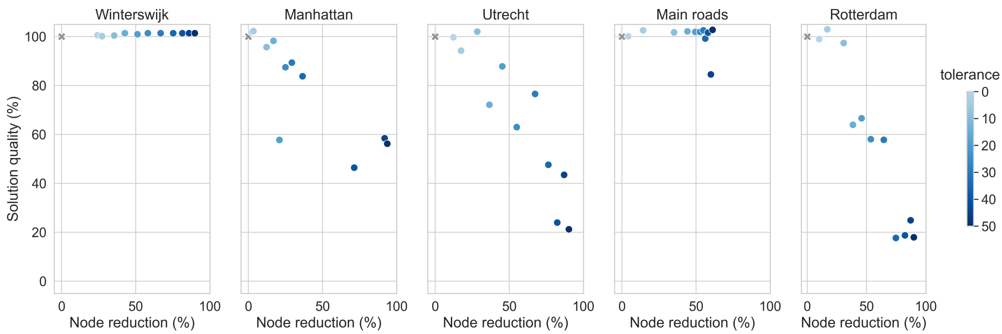
To maintain clarity, Figure 3.4 only shows the best solutions across seeds. The solution quality varies a lot between seeds, which improves with coarsening but still makes it difficult to observe clear trends. For all networks except Winterswijk, there are some outliers due to the optimization algorithm's sensitivity to the random seed, even across 10 seeds.

The impact of network size reduction on solution quality varies between networks (Figure 3.4a). For Winterswijk and the main road network, node consolidation has little to no effect on solution quality. The key interception positions and the paths from the starting positions are preserved. In Manhattan, we see a jump in node reduction once the tolerance exceeds the spacing between streets, which results in a large drop in solution quality. For Rotterdam, we see two of these drops: the first consolidations barely affect the solution quality, which then drops to approximately 60%, and later to 20%. For Utrecht, we see a more gradual but similarly dramatic decline in solution quality.

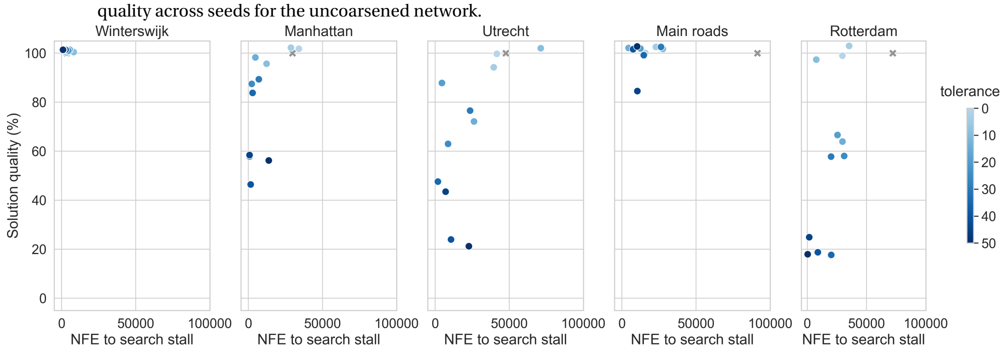
Figure 3.4b shows that reducing the number of nodes considerably decreases the number of function evaluations to search stall. The results for Winterswijk and the main road network are particularly interesting as the convergence speed is improved considerably without a loss in solution quality.

Table 3.4: Node reduction by node consolidation for the five case study road networks, relative to the uncoarsened networks, for a tolerance value of 5, 25 and 50.

	Node reduction (%)		
tolerance	5	25	50
Winterswijk	24.1	58.2	89.9
Manhattan	3.4	25.0	93.7
Utrecht	17.5	55.0	90.2
Main roads	14.3	52.7	61.2
Rotterdam	10.0	53.6	89.9



(a) Impact of node consolidation on the solution quality. The results are scaled to the best-found solution quality across seeds for the uncoarsened network.



(b) Impact of node consolidation on the convergence. The NFE to search stall is the NFE at which that seed reached 95% of its best-found solution quality.

Figure 3.4: Results of the node consolidation experiments, showing only the best result across seeds for each setting of the tolerance parameter. The grey crosses indicate the best result for the uncoarsened graph.

3.4.3. HEURISTIC COARSENING

BACKGROUND

Krishnakumari et al. (2020) propose a heuristic coarsening algorithm tailored to applications in transportation. The algorithm preserves key properties such as graph connectivity, shortest paths, and trip length distribution, making it a promising coarsening algorithm for fugitive interception.

The algorithm consists of four coarsening steps, repeated until either the maximum number of iterations is reached or the network cannot be coarsened further using the current settings.

1. Assign weights to the links in the graph. The weights can be based on properties relevant to the application, such as link length, road type, or speed.

2. Rank the nodes for removal. Instead of selecting nodes randomly, as is common in other coarsening algorithms, nodes are ranked deterministically for removal. This ensures reproducibility and reduces the computational time required for coarsening.

3. Contract and prune nodes. To avoid an excessive increase in the average node degree in the coarsened network (Geisberger et al., 2008), the algorithm applies a contraction criterion. Only nodes meeting this criterion are contracted. The tuning parameter 'threshold' (ρ) determines how strictly the criterion is followed, with higher values resulting in greater node reduction. Another parameter, pruning, removes dead ends, self-loops, and disconnected components.

4. Update the link weights. After contracting and pruning, link weights are recalculated for the coarsened graph, and steps 2-4 are repeated until the stopping criterion is reached.

IMPLEMENTATION

This study applies the heuristic coarsening algorithm from Krishnakumari et al. (2020), originally implemented in Matlab, which we have re-implemented in Python[1].

We conducted two sets of experiments. In the first set, we used the default settings, contracting nodes based on road type. These road types are crowdsourced in OpenStreetMap and range from 'pedestrian path' and 'bus lane' to 'motorway'. Nodes that serve as a connection between different types of roads (e.g., a highway and a residential street) are not contracted, assuming these nodes are important for the network's connectivity (Krishnakumari et al., 2020). In the second set of experiments, nodes are contracted based on their betweenness centrality. Nodes with a high betweenness centrality are preserved, assuming that these nodes are important for network connectivity and for fugitive escape routes.

The coarsening can be pre-processed, so the computation time of the coarsening algorithm does not affect the real-time performance of the decision support

[1] The Python implementation can be found at https://github.com/irene-sophia/HeuristicCoarsening

system. If the police starting positions, fugitive starting position, or escape nodes are removed during coarsening, the shortest paths from these positions to the coarsened network are added back to the network afterward. This post-processing step makes it possible to handle any incident location.

Both variants of the algorithm are tested with the same parameter settings as in Krishnakumari et al. (2020). We run the algorithm both for a single iteration and until completion, with thresholds set to either the minimum (0) or maximum (1000). We experiment with and without pruning.

RESULTS AND DISCUSSION

When coarsening the network based on road type, the node reduction is relatively limited (Table 3.5), but the solution quality declines quickly. Even for Winterswijk and the main road network, we see a large degradation of solution quality, while other coarsening algorithms had less problems with these networks. Pruning, in particular, causes a drastic decline in solution quality to 5 - 40 % of the original quality (Figure 3.5a). Again, the node reduction does considerably speed up the convergence, while obtaining poor solutions (Figure 3.5b).

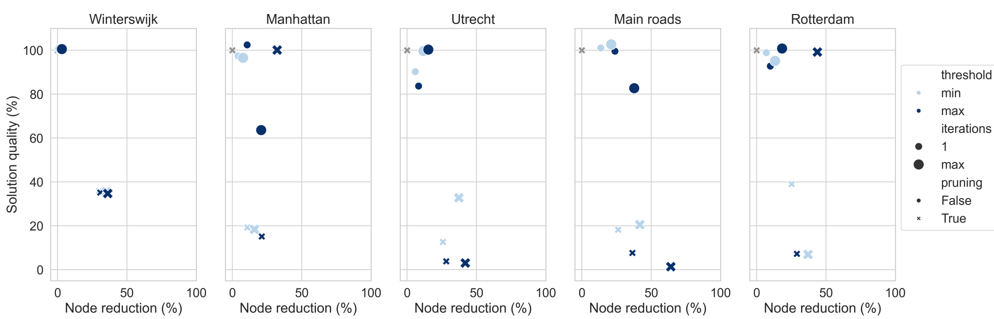
Using betweenness centrality improves the heuristic coarsening algorithm for fugitive interception (Figure 3.6). The solution quality for Utrecht, Rotterdam, and the main road network is affected little by the coarsening. For Winterswijk, the size of the road network is only reduced very little without pruning (Table 3.6). However, enabling pruning causes a sharp decline in solution quality. Similarly, pruning leads to very poor solution quality for Manhattan. Additional analysis shows that, while the original interception positions are largely preserved, the paths from the police starting positions to the interception positions are not. The node reduction does considerably speed up the convergence (Figure 3.6b).

Table 3.5: Node reduction by heuristic coarsening (type) for the five case study road networks, relative to the uncoarsened networks.

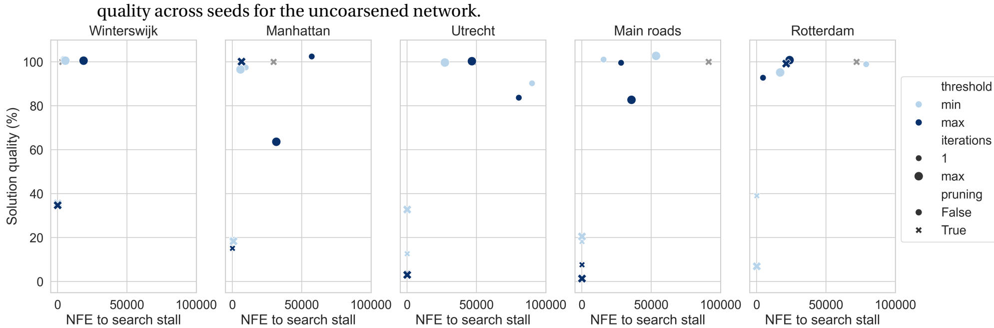
	Node reduction (%)			
pruning	0	0	1	1
iterations	1	max	1	max
threshold	0	1000	0	1000
Winterswijk	1.1	3.1	30.0	36.2
Manhattan	4.2	20.8	10.9	32.3
Utrecht	5.9	15.3	25.8	42.0
Main roads	38.6	55.7	47.5	74.5
Rotterdam	7.0	18.3	25.2	43.8

Table 3.6: Node reduction by heuristic coarsening (betweenness) for the five case study road networks, relative to the uncoarsened networks.

	Node reduction (%)			
pruning	0	0	1	1
iterations	1	max	1	max
threshold	0	1000	0	1000
Winterswijk	72.9	73.7	79.6	82.7
Manhattan	64.4	71.8	65.5	75.9
Utrecht	35.9	45.7	44.8	62.8
Main roads	54.4	71.6	56.8	83.6
Rotterdam	0.1	18.3	11.5	43.8

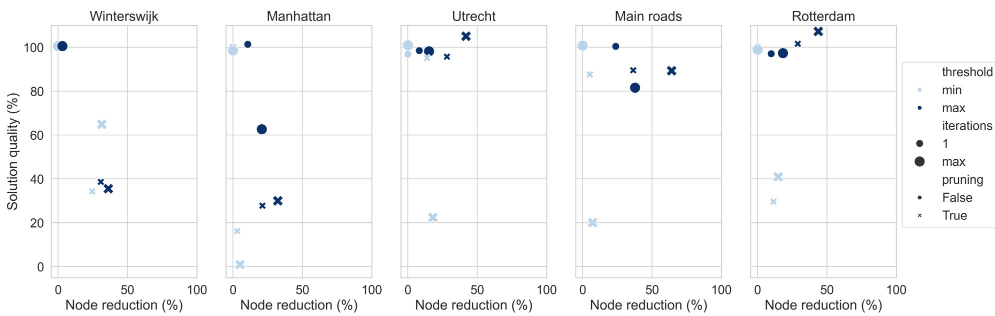


(a) Impact of heuristic coarsening on the solution quality. The results are scaled to the best-found solution quality across seeds for the uncoarsened network.

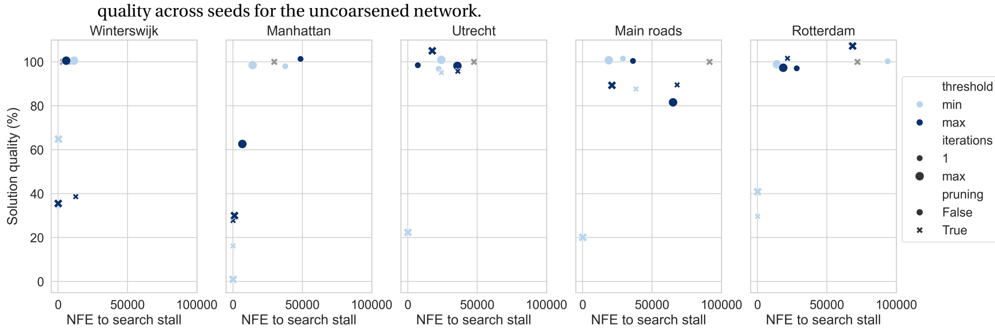


(b) Impact of heuristic coarsening on the convergence. The NFE to search stall is the NFE at which that seed reached 95% of its best-found solution quality.

Figure 3.5: Results of the heuristic coarsening (road type) experiments, showing only the best result across seeds for each parameter setting. The grey crosses indicate the best result for the uncoarsened graph.



(a) Impact of heuristic coarsening on the solution quality. The results are scaled to the best-found solution quality across seeds for the uncoarsened network.



(b) Impact of heuristic coarsening on the convergence. The NFE to search stall is the NFE at which that seed reached 95% of its best-found solution quality.

Figure 3.6: Results of the heuristic coarsening (betweenness centrality) experiments, showing only the best result across seeds for each parameter setting. The grey crosses indicate the best result for the uncoarsened graph.

3.4.4. ON-THE-FLY COARSENING

BACKGROUND

Instead of generic coarsening algorithms that allow for pre-processing, it is also possible to construct the road network on the fly for each optimization run. No important nodes or paths are lost, only the unimportant parts of the network for each specific case are removed. The on-the-fly coarsening method was developed as part of this paper. To our knowledge, no research exists that implements this network representation, though it could be a promising approach, especially for Flow Interception Problems.

IMPLEMENTATION

A new network is created from the simulated escape routes and the shortest paths from the police starting positions to any node on these escape routes. The network reconstruction takes a few seconds, depending on the number of nodes in the network (Winterswijk: 0.22 s, Utrecht: 3.74 s, Manhattan: 1.12 s, Main roads: 0.84 s, Rotterdam: 5.89 s).

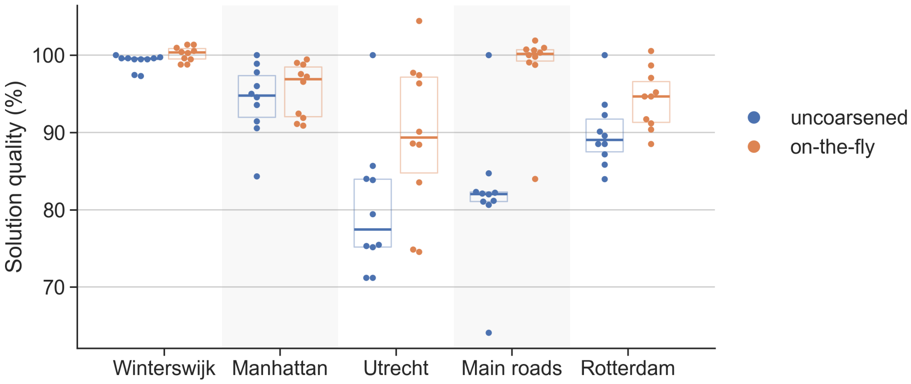
RESULTS AND DISCUSSION

Table 3.7 shows that removing unimportant parts of the network results in a considerable reduction in the number of nodes, ranging from 45.0 to 70.8%. This reduction improves the average solution quality across all networks (Figure 3.7a). By removing nodes that do not lie on escape routes or police paths, many possible combinations of police interception positions that do not lie on any fugitive route (with a solution quality of 0) are removed.

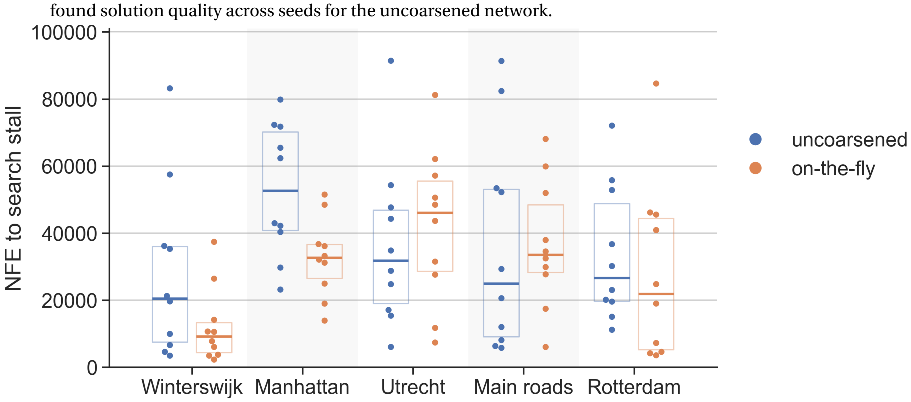
The number of function evaluations to search stall is lower for Winterswijk, Manhattan, and Rotterdam – networks with a smaller increase in solution quality from on-the-fly network reconstruction. In these cases, the node reduction primarily speeds up the search. On the other hand, for Utrecht and Manhattan, where the network reconstruction leads to a more substantial improvement in solution quality, the optimization requires more function evaluations. However, even in these cases, the optimization algorithm reaches a solution quality equivalent to the uncoarsened network earlier in the process. Additionally, the earliest NFE at which the search stalls (across multiple seeds) is lower, indicating a more efficient search despite the greater number of evaluations required for significant improvements. However, the minimum NFE at which the optimization algorithm finds a solution with the same quality as the uncoarsened network is lower. Additionally, across seeds, the earliest point at which the search stalls is also reached sooner, indicating that the network reconstruction not only improves solution quality but also speeds up convergence.

Table 3.7: Node reduction by on-the-fly network construction for the five case study road networks, relative to the uncoarsened networks.

City	Node reduction (%)
Winterswijk	70.8
Manhattan	45.0
Utrecht	52.3
Main roads	54.2
Rotterdam	52.9



(a) Impact of on-the-fly network construction on the solution quality. The results are scaled to the best-found solution quality across seeds for the uncoarsened network.



(b) Impact of on-the-fly network construction on the convergence. The NFE to search stall is the NFE at which that seed reached 95% of its best-found solution quality.

Figure 3.7: Results of the on-the-fly network construction experiments.

3.4.5. COMPARISON

Figure 3.8 combines the results, grouped by approach and by city. Appendix A presents a comparison of the results in tabular form. Evidently, pruning and, to a larger extent, on-the-fly network reconstruction reduce the size of the network while achieving the same or higher solution quality. The effectiveness of the other coarsening algorithms varies by network. Node consolidation and heuristic coarsening (based on betweenness centrality) perform well for Winterswijk and the main road network. Node consolidation shows a more gradual decline in solution quality, whereas heuristic coarsening – particularly when using the road type in the algorithm – shows a larger degradation, even at low node reduction. This is surprising since heuristic coarsening was specifically designed for transportation networks.

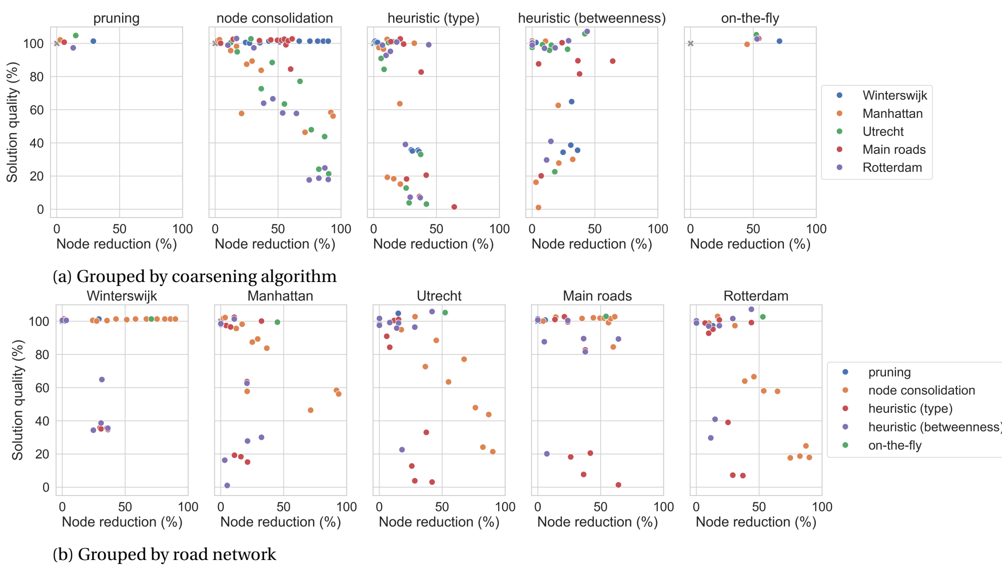


Figure 3.8: Comparison of coarsening algorithms. The solution quality is scaled to the best-found solution quality across seeds for the uncoarsened network.

Figure 3.9 shows the results of the timing experiments. The relationship between the number of nodes and the time per function evaluation generally follows a power-law trend. On-the-fly network reconstruction is an exception since many low-quality solutions are removed from the set of possible interception positions. For every function evaluation, the optimizer first checks whether a path exists between the police starting position and the candidate interception position. If that path exists, the length of the shortest path is calculated. That second step obviously adds to the computation time. In other words, a network with many infeasible solutions shows a shorter computation time, though the solution quality is poor. The experiments presented in Figure 3.7 have shown that on-the-fly network reconstruction significantly reduces the number of function evaluations to convergence.

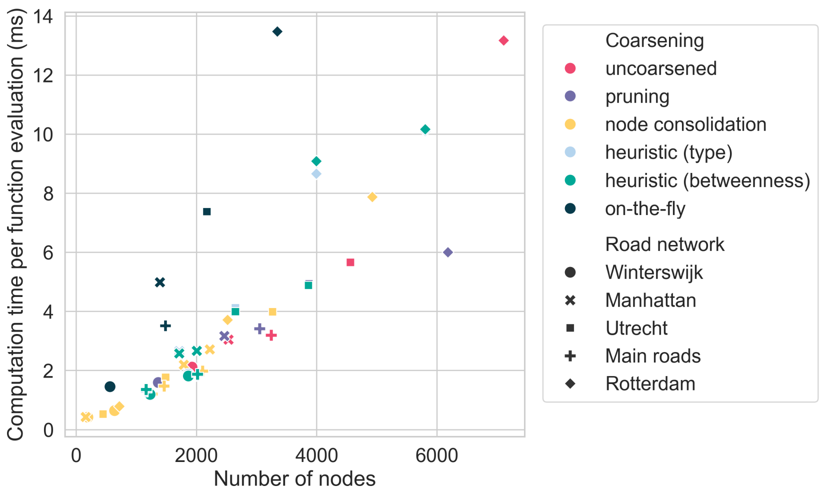


Figure 3.9: The computation time per function evaluation across different graph coarsening approaches and road networks.

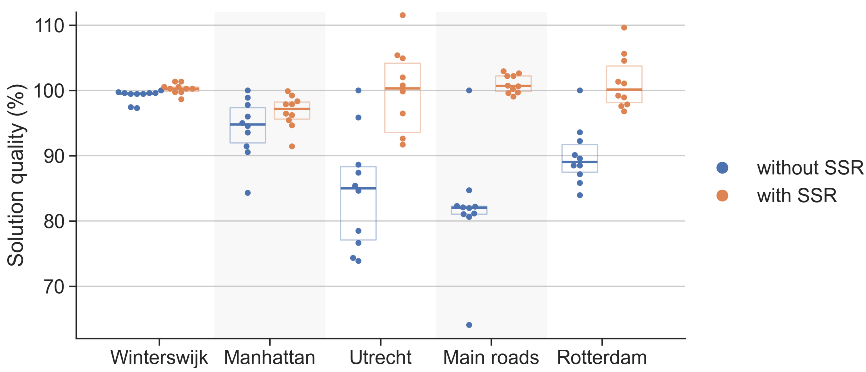
3.5. PROPOSED METHOD: SEARCH SPACE REPRESENTATION

The experiments show that pruning and on-the-fly network reconstruction effectively reduce network size while preserving solution quality. This, in turn, reduces the NFE to search stall and the time per function evaluation. In other words, the computation time is reduced in two ways.

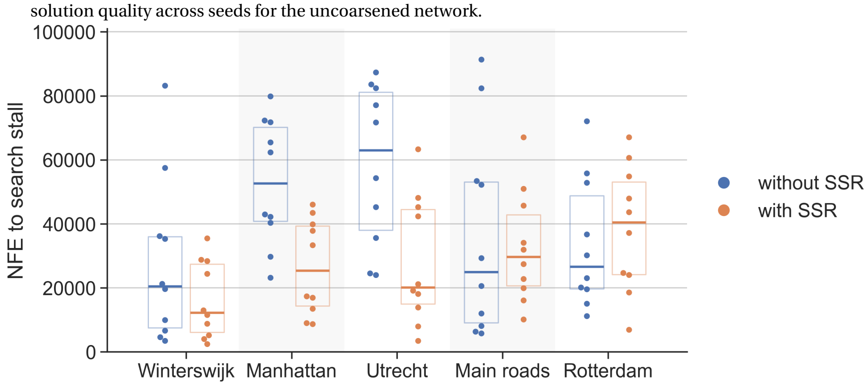
Drawing inspiration from Bode et al. (2019), we incorporate on-the-fly network construction in the representation of the search space, filtering out low-quality solutions. The remaining solutions are the combinations of police unit positions that a) are located on at least one escape route and b) can be reached by the respective police unit. introduces variability in the number of potential positions for different police units, depending on their initial locations. Compared to on-the-fly network reconstruction, the proposed Search Space Representation further reduces the size of the optimization problem by removing unreachable positions.

Figure 3.10 shows that this approach is generally effective in achieving a higher-quality solution using fewer function evaluations. For all case study road networks, the best-found solution quality increased, up to 12%. Especially for Utrecht, the Main roads and Rotterdam, both the average and best solution quality across seeds increased dramatically (Figure 3.10a). Increasing the average obtained solution quality improves the reliability of the optimization for the decision-maker. Since there are no possible solutions that have a solution quality of 0 (not intercepting any escape routes), the algorithm gets stuck less often in local optima and therefore converges to a solution with a higher quality. For Winterswijk, Manhattan and Utrecht, these high-quality solutions are also found in fewer function evaluations. For Rotterdam, the average number of function evaluations until the search stalls is higher, but the seed with the quickest convergence converges at a lower number of function evaluations than without the Search Space Representation (SSR). For the Main roads, the slower convergence is due to the significantly higher obtained solution quality: the earliest NFE at which the optimization algorithm finds a solution with the quality of the solution without the SSR is lower (Figure 3.10b). Across networks, the number of function evaluations required to obtain high-quality solutions is reduced to 3 000 -10 000 depending on the size and complexity of the road network. Considering that the time per function evaluation is 2-13 ms depending on the size of the network, this number of function evaluations is feasible for real-time decision-making.

Filtering the search space does not add to the overall computation time. In the optimization, the nodes are sorted on their proximity to the fugitive starting positions to speed up convergence (also in the uncoarsened case, Section 3.2). While the filtering step takes time, this is compensated because the number of nodes to be sorted is shorter. Depending on the network, this means that introducing the filtering step adds up to 0.3 seconds (Rotterdam) to reducing the computation time by 0.3 seconds (Utrecht), or does not impact the computation time (Winterswijk, Manhattan, Main roads).



(a) Impact of the proposed method (SSR) on the solution quality. The results are scaled to the best-found solution quality across seeds for the uncoarsened network.



(b) Impact of the proposed method (SSR) on the convergence. The NFE to search stall is the NFE at which that seed reached 95% of its best-found solution quality.

Figure 3.10: Results of the proposed method.

3.6. DISCUSSION

The effectiveness of graph coarsening algorithms for fugitive interception is dependent on the topology of the road network. Networks with dominant interception positions, like a highway network or Winterswijk, are relatively easily coarsened without degrading the solution quality. For other networks it proves difficult to find a general coarsening algorithm that both reduces the size of the network (and thus the computation time), while preserving the solution quality, i.e., the interception positions with a high probability of interception.

This research used a shortest-path model with noise to generate fugitive escape routes. Alternative models of fugitive behavior, such as avoiding busy roads, could affect the effectiveness of the coarsening algorithms that use preprocessing. The choice of escape nodes is also crucial; in this research, they are set at network boundaries like highway on-ramps or border crossings. If escape nodes were instead located at places like parking garages, the likely interception points would change, which could affect the effectiveness of the coarsening algorithms to different extents. In contrast, on-the-fly network reconstruction maintains the solution quality regardless of the fugitive escape routes. The solution quality is not affected by different models of fugitive behavior, but the reduction in computation time decreases when the number of nodes visited by the fugitive increases.

The data quality of open-source road networks influences experiments with graph coarsening. Since OpenStreetMap data is crowd-sourced, errors occur in network topology and attributes, such as road classification or speed limits. For example, in our research, we found a roundabout where one section was labeled as 'unclassified', while the rest was labeled as a 'residential road'. The heuristic coarsening algorithm that relies on the road classification to determine which nodes to contract, therefore produces incorrect results. Another example we encountered was a highway on-ramp that was mistakenly not connected to the main highway in the data. Such an error affects both the generation of escape routes and the suggested police paths to interception positions. In this case, the generated escape routes falsely suggested that the fugitive would not use the on-ramp. While we have corrected these mistakes, other errors likely persist in the data.

The Search Space Representation approach in this paper can be applied to other network-based optimization problems. In cases like search and rescue, where the location and paths of a lost person are uncertain (Koester, 2008), filtering out non-essential parts of the network and focusing on likely routes can significantly reduce computational complexity. For other network-based optimization problems, such as large-scale route planning, this method can help speed up computation and improve solution quality. Using the detailed network is only critical at the beginning (departure from the warehouse) and the end (delivery point). Preserving the full detailed graph is essential at these locations to ensure accurate routing, while coarsening the network in between could significantly speed up the optimization.

3.7. CONCLUSION

This paper compares four graph coarsening techniques for fugitive interception across five road networks. Pruning – the removal of dead ends and self-loops – seems to always be effective: it removes 2.7% to 29.1% of nodes (depending on the network), but these nodes are likely not relevant for fugitive interception. Other preprocessed graph coarsening algorithms can significantly reduce the number of nodes in the networks, but cause the solution quality to deteriorate significantly. Important interception positions and paths for the police units are often not preserved for these algorithms. In contrast, on-the-fly network reconstruction, where a new network is created from the escape routes and the shortest paths from the police starting positions to any node on these escape routes, improves the optimization. By removing poor-quality solutions, the optimization algorithm converges more quickly and results in higher-quality solutions.

Based on these results, we propose an approach incorporating on-the-fly graph reconstruction into the Search Space Representation in the optimization process. This allows for more flexibility, capable of handling different fugitive profiles and network structures. Search space representation improves the quality of the best solutions obtained by the optimization algorithm with up to 12%. Notably, the reliability of the optimization to find high-quality solutions is increased: the average obtained solution quality across seed increases by up to 24%. Meanwhile, the number of function evaluations required to obtain high-quality solutions is reduced to 5 000 -10 000 depending on the size and complexity of the road network, which is feasible for real-time decision-making.

The Search Space Representation approach in this paper can be applied to other network-based optimization problems, specifically search and rescue, and more generally to large-scale route planning.

4

THE EFFECT OF MODELS OF CRIMINAL BEHAVIOR ON POLICE INTERCEPTION

The previous chapters use a random walk model to generate escape routes. Meanwhile, the effectiveness of the police interception positions is dependent on the simulation model that generates the escape routes. Therefore, in this chapter, we investigate the effect of substituting the random walk model with models based on psychological theory.

This chapter is under review as: Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). The effect of models of criminal behavior on police interception.

The code and data associated with this chapter are available at: doi:10.4121/6d0f025b-967e-4007-9b6d-b920366d8f74

ABSTRACT

One of the tasks of police is catching fleeing suspects, where the police interception positions depend on the fleeing suspect's route choices. Various conceptualizations of route choice decision-making of fleeing suspects exist. However, we do not know the effects of these different models of fugitive behavior on the calculated police interception strategy. Therefore, we operationalize two models of route choice and implement these in a simulation. Police interception strategies are obtained by optimization. The resulting sets of routes and the calculated police interception positions are subsequently compared and interpreted. The experiments show that the different route-choice models result in different escape routes and, therefore, different calculated police interception positions. The differences are larger when the road network is complex and contains non-uniform obstacles. In other words, the robustness of the calculated police interception positions for each model largely depends on the network topology.

4.1. INTRODUCTION

In recent years, only 32% of reported crimes in the Netherlands resulted in the apprehension of a suspect (Moolenaar et al., 2023). In the US, this figure is 37% for violent crimes and only 12% for property crimes (Federal Bureau of Investigation, 2022). Furthermore, 85% of arrests are red-handed, meaning the suspects were caught in the act of committing a crime or immediately after committing a crime with incriminating evidence. The remaining 15% of arrests involve costly and time-intensive investigations (van Dijk et al., 2013). Increasing the number of red-handed arrests allows for more effective use of critical resources. Understanding the movement patterns of fleeing suspects and suggesting proper intervention positions for police units can limit the use of police resources and increase the chance of red-handed arrests.

Good interception positions are typically found at chokepoints in the road network, such as bridges and tunnels, where multiple roads converge. Mathematical optimization of the interception position relies on generating escape routes for the suspect (van Droffelaar et al., 2024b). The generated set of routes has to be complete in terms of network coverage and has to include the chokepoints. If not, the mathematical optimization model will not identify the most interesting interception points.

There are various ways to conceptualize the route choices of fleeing suspects to generate a set of escape routes (Sava et al., 2016; van Gelder, 2013). Without knowledge of their underlying decision-making process, the routes may resemble a random walk through the road network. In contrast, if we had complete information on the suspect's characteristics and decisions, there would be a single deterministic route. In practice, we have incomplete information, where we have some understanding of route choices but not all, leading to a heuristic implementation of the route choice model of a fugitive.

Many theoretical studies implement a random motion for the fleeing suspect (Borie et al., 2013; Sava et al., 2016). Explicitly encoding behavior through decision rules could lead to more effective interception strategies (Simard et al., 2021). Therefore, the central question in this paper is: what is the effect of different models of fugitive behavior on the calculated police interception strategy? To answer this question, we conceptualize and operationalize two modes of fleeing suspect route choices. We compare the resulting sets of routes and the optimized police interception positions. Finally, we evaluate the effectiveness of the police interception positions for different route generation models.

The following section discusses the related literature on modeling behavior in interception problems and, specifically, modeling fugitive route choice behavior. Section 4.3 describes the methods used in the paper, including a description of the case studies, the models, and the optimization algorithms. The subsequent section details the obtained results. Possible threats to the validity of the results are discussed in Section 4.4.4, and we share our conclusions in Section 4.5.

4.2. MODELING BEHAVIOR

Different fields use different terminology when talking about interception. Game theory and mobile robotics refer to pursuit-evasion games, where the terms *pursuer*, *evader*, or *target* are used (Chung et al., 2011). Mathematical optimization refers to similar problems as search problems or interception problems, depending on the objective function (Alspach, 2004). Game theory, mobile robotics, and mathematical optimization take a theoretical approach, reflected in the abstract naming. In contrast, empirical research on police interception refers to *police units* and *suspects* or *fugitives* (Dewinter et al., 2022). In this paper, we will follow the terminology belonging to each field when discussing related literature and use the terms *police unit* and *fugitive* in the case study.

The following subsections discuss how targets are modeled in interception problems and explore various conceptualizations of criminal behavior. Each section first outlines the relevant literature and subsequently discusses its relation to this paper and resulting modeling choices.

4.2.1. MODELING BEHAVIOR IN INTERCEPTION PROBLEMS

Various methods exist to model targets' behavior in the search, pursuit, and interception literature. Inspired by the mindmap on search in mobile robotics in Chung et al. (2011), we use the structure depicted in Figure 4.1. The Figure provides options for motion, behavior, and availability of information, which are subsequently discussed.

Motion Pursuit-evasion games and search problems are solved for both stationary and mobile targets. For a comprehensive overview of stationary target interception, see Stone et al. (2016). Most other papers discussed in this section treat

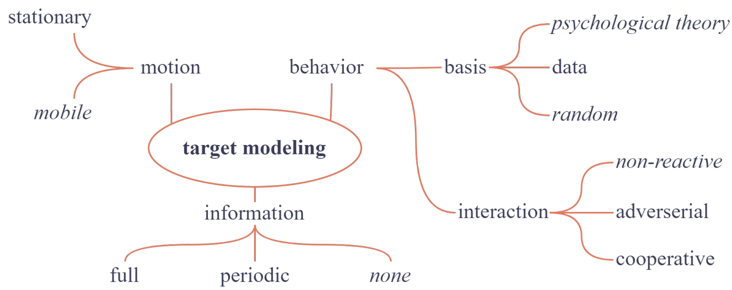


Figure 4.1: A graphical overview of the dimensions of target modeling in interception problems. The subset operationalized in this paper is italicized.

4

mobile targets, as that is the focus of this paper. For a survey of graph-based search of mobile targets, see Alspach (2004).

Behavior Many studies implement a random motion for the evader. This approach is usually adopted when there is no understanding of the underlying behavior (Simard et al., 2021) or when the focus is purely on optimizing search strategies (Borie et al., 2013). In the search and rescue literature, random motion is one of the modes of behavior found in empirical data (Hashimoto et al., 2022; Koester, 2008). A lost person trying to find their way may move so erratically that it is close to a random walk. On the other hand, various models, often agent-based, encode rules of behavior of lost persons (Hashimoto et al., 2022) or evaders (Giardini et al., 2023). These models can be either based on historical data or grounded in psychological theories. A small, simple set of decision rules can generate complex emergent patterns (Epstein et al., 1996).

Depending on the intended application, evaders are either modeled to be non-reactive, cooperative, or adversarial. Non-reactive evaders move independently of the moves of the pursuer, either randomly or according to some predetermined pattern. This behavior occurs, for example, when the evader does not have any information on the whereabouts of the pursuer. For search and rescue, some modes of behavior of lost persons are non-reactive. Other lost person modes are cooperative: they move to places in the network where they assume that the searchers are likely to look or where they can signal their presence to the searchers. Lastly, adversarial evaders actively evade capture by anticipating and reacting to the pursuer's moves. Adversarial evaders are, for example, found in hide-and-seek (Baker et al., 2020) and some pursuit-evasion games.

Information Information about the evader's whereabouts can help to find more effective pursuer search strategies and adapt to the evader's strategy. On the other hand, information about the pursuer's location is crucial to optimize an adversarial evader's route to escape capture. Information (in either direction) can either be full (Borie et al., 2013), based on proximity (Kehagias et al., 2014), based on visibility (Bopardikar et al., 2008) or using a sensor network (Schenato et al., 2005).

In fugitive interception, there is very little exchange of information about each other's positions: the fleeing suspect does not know the locations of the intercepting police units, and vice versa. Spotters, traffic cameras, or phone calls from concerned citizens can provide sparse information about the fugitive's location. Furthermore, very little historical data is available, meaning that data-driven approaches are not applicable. Even with more available data, a model based on historical data would be vulnerable to survivorship bias (as we only have information on successful cases where the suspect was caught) and historical bias (as the data may no longer reflect the current *modi operandi*). Therefore, we opt to build generative models grounded in psychological theory for this application.

The next section explores psychological theories that may be leveraged to build generative models for simulating fugitive escape routes.

4.2.2. MODELING CRIMINAL BEHAVIOR

Little is known about the decision-making of suspects fleeing a crime scene. However, we can draw from the broader field of criminology to generate insights.

The rational choice theory of crime is a leading framework in criminology to understand criminal behavior (Cornish & Clarke, 1986). The central thesis is that individuals make rational decisions to engage in criminal activities after weighing the potential costs and benefits. van Gelder (2013) goes beyond Rational Choice Theory, proposing two modi of criminal decision-making. Besides the 'cool' rational mode, calculating costs and benefits, there is a 'hot' mode that evaluates options in a more intuitive way. This concept has a long history and is more generally referred to as bounded rationality (Simon, 1982), as heuristic versus analytical (Evans, 1984), as intuitive–experiential and analytical–rational (Epstein et al., 1996), as system 1 and system 2 (Stanovich & West, 2000), as cognition and emotion (Kahneman, 2003; Kennedy, 2012), and as sense and sensibility (Vanhée & Borit, 2023). Simon (1982) explains bounded rationality as "agents use simple rules based on local information - not global information with infinite computing power." Similarly, a criminal in van Gelder (2013)'s 'hot mode' responds to situational characteristics. Premeditation, i.e., preparing a crime, can result in more 'cool' behavior, compared to street offenders in hedonistic contexts (Shover & Hochstetler, 2005).

In this paper, we operationalize these 'cool' and 'hot' modes using the relevant behavioral factors found by Tutuarima (2023). Her work synthesizes general route choice literature, specifically evacuation literature, and interviews with domain experts. We implement the following factors:

- Camera avoidance: a prepared suspect avoids passing by cameras that the Police can access.

- Obstacle avoidance: traffic lights, roundabouts, bridges, and tunnels are — to a varying extent — less attractive due to unpredictability and becoming potential choke-points (Moussaïd et al., 2011).

- Number of lanes and maximum speed: a stressed suspect prefers a higher number of lanes and maximum speed so they can get away from the crime location as quickly as possible (Kempenaar, 2022).
- Inertia: a stressed suspect is more likely to continue on the road they are on (Alós-Ferrer et al., 2016; Meneguzzer, 2023).

Relevant personal attributes (risk aversion and familiarity with the road network) and contextual factors (time of day and crime location) are not considered in this research. Interaction with other traffic is also not considered. Instead, we develop simpler, generalized models that generate a wide range of options, minimizing the risk of dual use of the route generation models by bad-faith actors.

4.3. METHOD

4.3.1. MODELS

We implement two models of fleeing suspect route choices ('cool' and 'hot')[1]. Both models aim to reach a predetermined set of escape nodes from the incident location, but the road preferences vary. Although unrealistic for fugitive behavior, a random walk model is used as a benchmark because it is commonly used in literature, and it is a quick method of generating a broad set of routes.

COOL MODE

The well-prepared, cool model prioritizes avoiding cameras to avoid detection. Traffic lights are avoided because they may add an unpredictable delay and traffic. Roundabouts are avoided if possible because they are difficult to navigate and oversee. Tunnels and bridges are avoided because they create a lock-in, they are difficult to oversee, and they are seen as a likely police position. In practice, all major tunnels and bridges are overseen by cameras at their entrance and exit, leading to a cumulative perceived delay of 35 or 65 seconds.

HOT MODE

The stressed, ad-hoc, hot model prioritizes avoiding traffic lights, which mimics turning when encountering a red light. Roundabouts and bridges are, analogous to the cool model, avoided if possible because they are difficult to navigate and oversee. The hot model avoids tunnels more due to feeling trapped. Additionally, the hot model prefers roads with more lanes and higher speed limits for easier maneuverability and perceived faster escape.

DIRECTED RANDOM WALK

The 'hot' and 'cool' models are contrasted with a directed random walk. This method is commonly used and computationally cheap. Like the other models, the random walk starts at the location of the incident, as defined in Section 4.3.4. At each intersection, the fugitive chooses the next node to travel to, following

[1] All data and code can be found at: https://github.com/irene-sophia/fug_behavior

Table 4.1: Operationalization of the behavioral models; times and factors indicate the perceived delays by the fugitive

Perceived delay	'Cool' model	'Hot' model
Camera	+30 s	—
Traffic light	+10 s	+20 s
Roundabout	+5 s	+5 s
Tunnel	+5 s	+10 s
Bridge	+5 s	+5 s
Lanes	—	1: ×1.2, 2:×1, ≥3: ×0.8
Speed limit (km/h)	—	≤30: ×1.2, ≥30:×1, ≥50: ×0.9, ≥80: ×0.8

a stochastic process where each neighboring node has an equal probability of being chosen. The fugitive does not turn around unless the node only has one neighboring node (i.e., a dead end).

Each factor identified as important for fugitive route choices is assigned a weight representing its 'added travel time'. In other words, it defines how much shorter a route must be to be more attractive than a longer route that avoids the obstacle. The order and approximate values follow from interviews with domain experts. Table 4.1 presents the perceived delays of each obstacle. Locations of cameras and obstacles are obtained from open data[2] and OpenStreetMap.

After adding the perceived delays to the travel time of the respective links of the road network, the perceived best routes are generated for each model. Next, a noise of either 2% or 5% (Cool mode) or 5% or 10% (Hot mode) is added to routes, meaning that the suspect takes a wrong turn every X% of the intersections. After a wrong turn, the new best path is determined from their next position. This noise accounts for three factors: (1) simulating human error, especially in stressful situations; (2) accommodating adjustments for unexpected obstacles like red lights; and (3) accounting for factors not explicitly modeled. As a result, we observe a distribution of routes around the optimal paths.

4.3.2. OPTIMIZATION

We model the optimization problem as a variation on the Flow Interception Problem (Berman et al., 1992; Hodgson, 1990). The target position for each police unit is optimized to maximize the number of intercepted routes. A route is intercepted if (1) the route passes a target position of a police unit and (2) that police unit can reach its target position before the escape route passes through. The optimization problem is NP-hard, meaning that solving real-world instances involves computation times of years. Optimizing the route of each police unit, where they can intercept the fugitive at any intermediate time, or dynamically reacting to incoming

[2]Camera locations were obtained from (1) Utrecht, the Netherlands: https://data.utrecht.nl/dataset/cameraregister-utrecht, public security cameras, (2) Manhattan, New York, USA: https://banthescan.amnesty.org/decode/, traffic cameras, (3) Winterswijk, the Netherlands: https://www.politie.nl/informatie/locaties-cameraplan-anpr-126jj-sv.html, traffic cameras.

information about the fugitive's whereabouts would further increase the complexity and required computation time.

We solve the optimization problem using a genetic algorithm supplemented with the auto-adaptive framework from Borg, which co-evolves the probabilities of the evolutionary operators used for population adaptation based on their relative success in finding fitter offspring (Hadka & Reed, 2013). van Droffelaar et al. (2024b) show that this optimization approach quickly finds near-optimal solutions. To further ensure the quality of the solutions, the algorithm is run with five seeds for 20,000 function evaluations, only preserving the best solution.

4.3.3. DESIGN OF EXPERIMENTS

We examine the effect of different fleeing suspect route choice models on the resulting simulated escape routes and the calculated police interception positions. Finally, we cross-evaluate the calculated interception positions on different sets of simulated escape routes. For this, we use the following design of experiments, graphically displayed in Figure 4.2:

1. A set of 1000 routes is generated by looping through all escape nodes and generating the shortest route (based on the adjusted perceived travel time).

2. These routes form the input to a pyDSOL discrete-event simulation model . An entity is created for each route. In the pyDSOL model, each entity follows its predetermined route unless it takes the wrong turn (determined by the 'noise' parameter). A wrong turn is a random choice of the neighboring links, excluding the planned and previous ones. Each combination of model and noise generates a set of routes constituting the first set of results.

3. These routes are the input of an optimization model that determines the positions for a set of police units that maximizes the number of intercepted routes. The calculated interception positions form the second set of results.

4. The positions are evaluated against different models to test their robustness. In other words, we determine the number of intercepted routes resulting from a different model using the positions optimized for the original model.

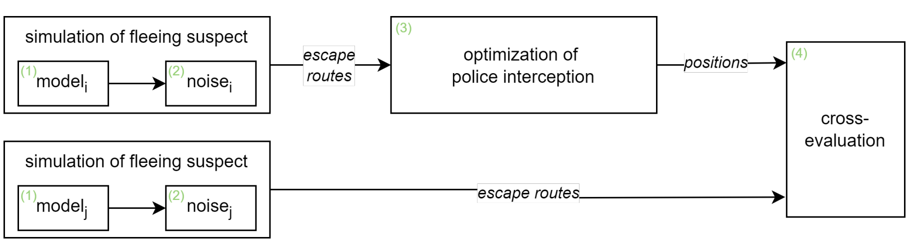
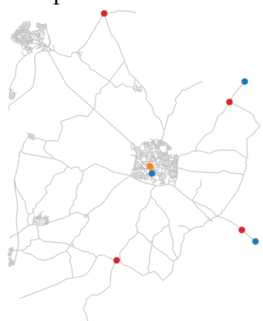
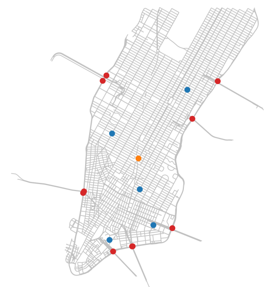


Figure 4.2: Graphical overview of the design of experiments. The green numbers in parentheses refer to the numbered items in Section 4.3.3.

Table 4.2: Case study road networks used in this study. Incident locations are marked in orange; escape nodes in red, and police starting positions in blue.

Utrecht	Winterswijk	Manhattan
Typical European city	Rural area	Modern, grid city
Incident: city center	Incident: city center	Incident: Union Square
Escape: the highway	Escape: the border	Escape: off the peninsula
		

4.3.4. Road Networks

The experiments are performed for three case studies (Table 4.2). Utrecht, the Netherlands represents a typical European city with a historical center surrounded by modern neighborhoods. Winterswijk, the Netherlands represents a rural area with sparse roads surrounding a town. Escaping by crossing the border to Germany is possible in the north, east, and south. Manhattan, New York, USA represents a modern grid layout city with traffic lights and cameras at most intersections. The police start locations are the local police stations.

4.4. Results & Discussion

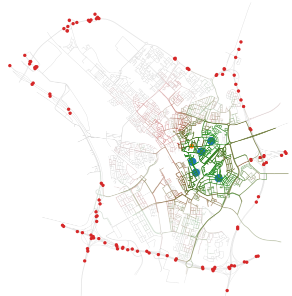
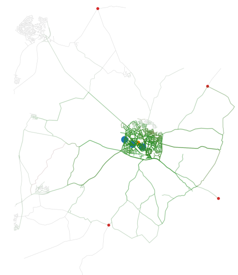
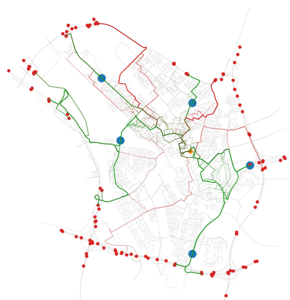
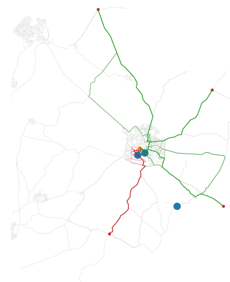
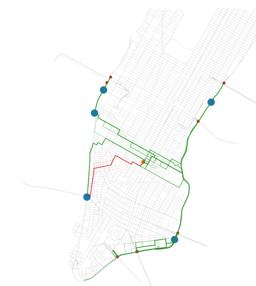
The simulation results in Table 4.3 display the escape routes in red and green, indicating whether they are intercepted by the calculated police interception positions (blue dots). The following sections discuss the results: first, the escape routes, then the calculated positions, and last, the robustness evaluation.

4.4.1. Simulated Escape Routes

The escape routes resulting from the various models are presented in Table 4.3. Below, we discuss the results, first in general and then individually for each city.

In general, the characteristics of the simulated escape routes depend on the underlying road network. However, directed random walks are an exception, as they tend to stay near the starting point and rarely reach the designated escape nodes. Additionally, the Cool and Hot models show different road preferences across all networks. With increasing noise, routes spread out more, visiting more nodes, but the differences between Cool and Hot persist.

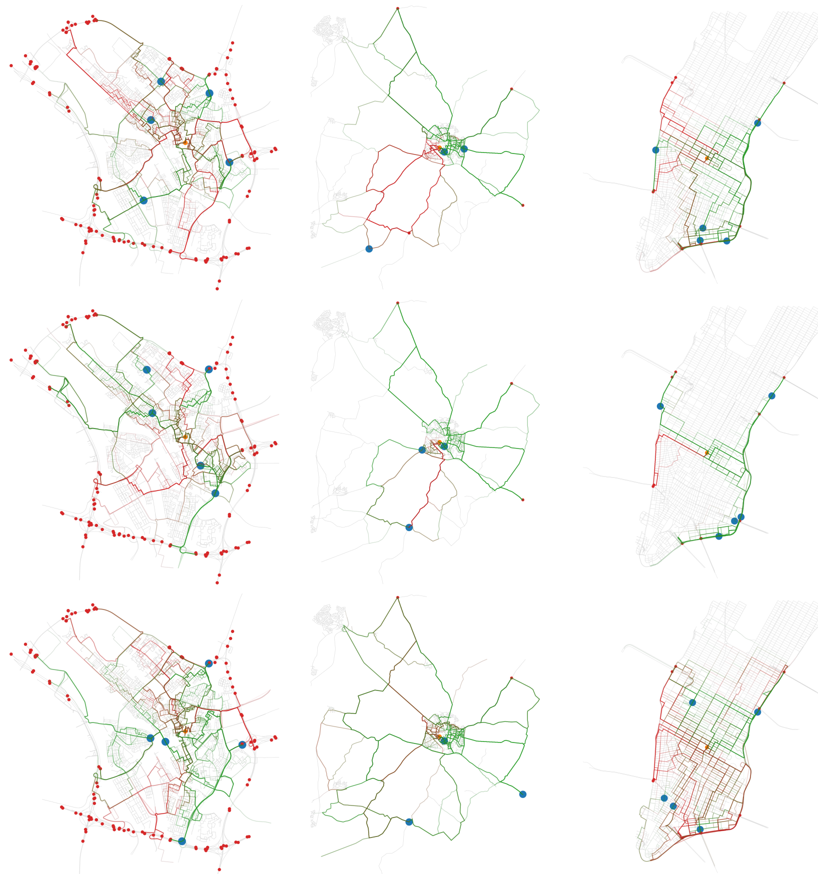
Table 4.3: Resulting escape routes, with intercepted routes in green, not intercepted routes in red, and calculated interception positions in blue.

Model	Utrecht	Winterswijk	Manhattan
Random walk			
Cool, 2% noise			

Cool, 5% noise

Hot, 5% noise

Hot, 10% noise



Utrecht shows several equivalently attractive roads near the incident in the city center. A few major arteries fan out from the city center towards the escape nodes, but the chosen arteries differ between the Hot and Cool models. For instance, comparing the southwest of Utrecht for the Cool model with 2% noise and the Hot model with 5% noise shows clear differences. In Winterswijk, only a few roads lead to the designated escape nodes, resulting in similar Hot and Cool graphs, with noise being the main distinguishing factor. In Manhattan, low-noise models gravitate towards the major riverside roadways due to the relative absence of traffic lights and cameras. The preferred roads differ between the Hot and Cool models. The Hot model is relatively less sensitive to noise (comparing Cool and Hot with 5% noise) since there is a traffic light at virtually every intersection, and cameras are slightly more sparsely distributed. Therefore, when the Hot model takes a non-optimal turn due to the modeled noise, making a U-turn is more attractive than taking a detour. At higher noise (10%), the model is pushed out of that equilibrium, and the routes spread out widely due to the grid layout with nearly euivalent.

In summary, the cool and hot models result in different road preferences. The specific characteristics are dependent on the network features and topology.

4.4.2. CALCULATED INTERCEPTION POSITIONS

The calculated positions are shown as blue dots in Table 4.3. The positions are optimized to maximize the number of intercepted routes. Note that a route is only intercepted if the associated police unit can reach the position before the route passes it. The initial police positions are shown in Table 4.2.

Since the random walks remain stuck near the incident, the calculated interception positions are also near the incident, at very different positions than other models. The calculated positions for Utrecht are along major city roads, further from the incident. Yet, the specific positions vary across models. One consistently calculated position, located centrally in the East, proves effective regardless of the suspect's model. Similarly, in Winterswijk, a position near the incident remains constant across all models: one nearby police unit can quickly intercept many routes towards the north and east of the network, while others disperse across the network depending on the model used for the suspect. In Manhattan, the calculated interception positions are consistent across Cool and Hot models and are concentrated along the major roadways along the river.

In summary, the Cool and Hot models result in different calculated police interception positions. The specific positions are dependent on the road network features and topology. Some consistently well-performing positions are found.

4.4.3. ROBUSTNESS EVALUATION

We cross-evaluate the effectiveness of calculated police interception positions for each model (i) by examining the number of intercepted routes generated by other models (j). The results, shown in Figure 4.3, are scaled to the number of intercepted routes using the same optimization and evaluation model ($i = j$). Therefore, each row indicates the robustness of the calculated positions across different models of suspect route choices. Negative values (pink) indicate that the inter-

(a) Utrecht (b) Winterswijk (c) Manhattan

Figure 4.3: Results of the robustness evaluation.

ception positions calculated based on the optimization model perform worse on the routes generated using the evaluation model compared to the routes generated using the optimization model (when $i = j$). A positive score (blue) means that relatively more routes are intercepted, suggesting that the routes of the evaluation model are less spread out than those of the optimization model.

Irrespective of the road network, the random walk model generates distinctly different routes, so positions based on this model perform poorly on other sets of routes and vice versa. An exception is Winterswijk, where any set of calculated positions performs well for random walk routes, likely due to their proximity to the incident location, where the random walk routes concentrate.

The bottom row and rightmost column display the superset of routes, combining the Cool 2%, Cool 5%, Hot 5%, and Hot 10% routes. The results confirm that interception positions calculated for this superset are generally robust across models. While they do not significantly outperform other optimization models, they offer a reliable approach if the mode of behavior is unknown.

The heatmap of Utrecht (Figure 4.3a) shows that positions calculated based on the set of escape routes generated using one model do not perform well on another set of routes. Numerous potential escape nodes and equally attractive roads leading to them result in considerable differences in calculated positions across models. Notably, the Cool 5% model shows higher robustness compared to others. The Winterswijk heatmap (Figure 4.3b) is relatively homogeneous, with most values around 0. This can be explained by, first, the generally good interception position in the city center and, second, the limited number of roads that lead to the escape nodes. The Manhattan heatmap (Figure 4.3c) also shows a somewhat uniform pattern. Generated escape routes converge on riverside roadways, leading to a concentration of calculated positions. However, the Hot 10% model is an exception due to its broad set of escape routes covering most nodes in the road network. Consequently, the calculated positions based on this model are less concentrated on the riverside roadways. This results in poor performance when applying positions from the Hot 10% model to other models and vice versa.

In summary, the robustness of the calculated police interception positions for each model largely depends on network topology. In compact networks with few escape nodes and distinct attractive roads leading to them, funnels emerge that form good interception positions. However, in more uniform road networks, understanding the suspect's route choice is crucial for successful interception.

4.4.4. DISCUSSION

This paper shows that the effectiveness of police interception depends on the route choice model of the fugitive. The specific characteristics of escape routes and interception positions largely depend on the case study network. Therefore, further research should explore more types of road networks and identify network characteristics that consistently lead to effective interception positions.

This paper presents a first attempt to operationalize conceptual models of fugitive escape route decision-making. Additional state information that might influence fugitive behavior, such as the type of crime (ram raid, robbery, pickpocketing, etc.) and the traffic situation, can be added. Additionally, the model of behavior might switch during the escape, for example, shift to 'cool' after some time or shift to 'hot' when unexpected things occur. Expert interviews can help to determine relevant characteristics to add to the choice model. However, there is a limit to predictability: the police do not know the exact psychological state of the fleeing fugitive, and empirical data on fugitive routes does not exist. Additionally, limited computation time for real-time decision support constrains the complexity of the models that can be used.

4.5. CONCLUSION

Knowledge of the specific route choice model of the fleeing suspect is critical for finding effective interception positions in complex networks with non-uniformly distributed features and obstacles. This paper conceptualizes and operationalizes three models of fleeing behavior to examine the resulting routes, calculated police interception positions, and the robustness of the models. We show that

- A random walk model - often used to simulate fleeing suspects in interception problems - leads to distinctly different escape routes and, therefore, calculated interception positions compared to models based on psychological theory. Therefore, a random walk model is unsuitable for decision support in real-world police interception.

- Despite their similarities in implementation, the Cool and Hot models result in different simulated escape routes and, therefore, calculated police interception positions. The differences are larger when the road network is complex and has non-uniformly distributed obstacles.

- The calculated interception positions are robust to different models of a fleeing suspect when the road network is either (1) relatively simple with few roads leading to the escape nodes, or (2) when police units can quickly reach intersections close to the incident, or (3) the positions of the escape nodes create a funnel where escape routes converge.

Further research should focus on extending the library of plausible models of fleeing suspect route choices based on data and interviews with domain experts. Additionally, exploring a wider range of road networks should identify network characteristics that consistently lead to effective interception positions.

5

TIMELY ADAPTIVE STRATEGIES FOR FUGITIVE INTERCEPTION

The previous chapters have considered a static optimization problem, where the police interception positions are calculated once. In reality, information about the fugitive's location becomes available during the interception attempt. This chapter evaluates promising solution approaches that adapt to the incoming information.

This chapter is under review as: Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). Timely adaptive strategies for fugitive interception.

The code and data associated with this chapter are available at: doi:10.4121/fa299948-661f-4003-a4c1-a4f3a6bb2809

ABSTRACT

The police need operational support to intercept fleeing suspects. The unpredictability of the fleeing suspect and not knowing if, where, and when a traffic camera detects the fugitive introduce uncertainty to the problem. Moreover, there is clear path dependency: sending police units in a certain direction constrains their possible rerouting in the future. In other words, there is a trade-off between the flexibility to react to new information and the timeliness of decisions.

Models can support the decision of where to position police units. Traditional stochastic optimization methods for solving fugitive interception do not account for the path dependency. Two promising adaptive approaches are Policy Tree Optimization and Direct Policy Search. However, these solution approaches have not been applied to fugitive interception, which has a rugged fitness landscape and requires the solution to be calculated in real time to be relevant to decision-makers. Therefore, this study evaluates the performance of policy tree optimization and direct policy search on the fugitive interception problem for various road networks.

5.1. INTRODUCTION

Police interception of a fleeing fugitive is complex due to the unpredictability of the fleeing fugitive, the many possible interception strategies, and limited decision-making time (Mehlbaum et al., 2014; van Dijk et al., 2013). Models can support decision-making by suggesting promising interception positions. Van Droffelaar et al. (2024b) show the feasibility of sequential simulation-optimization for real-time decision support for fugitive interception. They use simulation models to generate an ensemble of plausible routes for the fleeing fugitive and metaheuristic optimization methods to find interception positions for the police with a high probability of interception. However, these positions are static and do not react to updated information about the whereabouts of the fugitive.

The police can access traffic cameras with automatic number plate recognition (ANPR) software. Additionally, they may receive calls from concerned citizens regarding abnormal or dangerous behavior. These information sources can help to narrow the search for a fleeing fugitive and, therefore, increase the probability of interception. Using the information to increase the probability of interception is not straightforward due to the unpredictability of the behavior of the fugitive and not knowing if, where, and when a sensor detects the fugitive. Given the known positions of sensors that might detect the fugitive, both detections and lack of detection rule out possible escape routes. Moreover, there is clear path dependency: sending police units in a certain direction constrains their possible rerouting in the future. In other words, there is a trade-off between the flexibility to react to new information and the timeliness of decisions.

The literature presents various methods for dealing with incoming information in simulation-optimization (Henrichs et al., 2022). Yet, it is unknown which approach is most effective for fugitive interception problems. Traditional stochastic online optimization methods, such as Periodic Re-Optimization (Psaraftis, 1980), do not account for uncertainty or path dependency. On the other hand, most techniques developed for adaptive decision-making under uncertainty are developed for long-term planning problems and require ample time for analysis and intermediate input from decision-makers. Direct Policy Search (Giuliani et al., 2016; Koutsoyiannis & Economou, 2003) and Policy Tree Optimization (Herman & Giuliani, 2018) are developed for optimal control under uncertainty and may be suitable for real-time decision-making. However, timely calculation of the solution is essential to support police interception operations in real time. Direct Policy Search optimizes a policy, described by the parametrization of Radial Basis Functions, that maps the system's state (in our case, ANPR input) to control actions (Giuliani et al., 2016; Rosenstein & Barto, 2001). Policy tree optimization optimizes a binary decision tree that delineates what actions should be taken under what conditions (i.e., ANPR input) (Herman & Giuliani, 2018). Policy Tree Optimization is, therefore, less expressive than Direct Policy Search but yields a more interpretable output.

We implement and compare three promising adaptive approaches: Periodic Re-Optimization, Direct Policy Search, and Policy Tree Optimization for the fugitive interception problem with information updates. We evaluate the solution approaches on three types of networks: (1) a demonstration network to observe how the various approaches treat incoming information; (2) a 2D 10x10 grid with equidistant vertices; (3) three distinct city road networks to assess the generalizability to real-world decision contexts. The online approaches are contrasted with one-shot optimization, where the interception positions are not updated based on new information, to evaluate the added value of information.

Section 5.2 describes the literature related to fugitive interception (Subsection 5.2.1) and online optimization (Subsection 5.2.2). Section 5.3 describes the formalization of the fugitive interception problem, the implementation of the solution approaches used in the paper, and the experimental setup. Sections 5.4.1, 5.4.2 and 5.4.3 respectively discuss the results for the test network, a grid network, and the city road networks. The implications of the results are discussed in Section 5.5, and, finally, we share our conclusions in Section 5.6.

5.2. RELATED LITERATURE

5.2.1. FUGITIVE INTERCEPTION

This paper considers the fugitive interception problem, where the positions of police units are optimized to maximize the probability of intercepting a fleeing fugitive on a road network. The problem is formalized as a variation on the Flow Interception Problem, introduced by Hodgson (1990) and further developed by Berman et al. (1992). The original Flow Interception Problem aims to maximize the intercepted flow by a designated number of facilities, for example, consumers encountering at least one facility during their journeys. Gendreau et al. (2000) expanded the model by introducing gain coefficients a_{rv} for each vertex v in route r, explicitly connecting the objective value to flow values. In a subsequent adaptation, Tanaka and Kurita (2020) modified the Flow Interception Problem to handle probabilistic interception and reward earlier interception. For fugitive interception, the generic Flow Interception Problem is extended with a reachability constraint that ensures the police unit can reach an interception vertex before the fugitive. The generic Flow Interception Problem is NP-hard, meaning it cannot be solved in polynomial time (Boccia et al., 2009). The problem, however, requires extremely fast responses since the police have, at most, a few minutes to determine where to send available police units. Therefore, (van Droffelaar et al. (2024b)) show that metaheuristic solution approaches yield near-optimal solutions and demonstrate and real-time applicability.

This study extends the fugitive interception problem to consider incoming information that informs the positions of the police units. To this end, we compare adaptive solution approaches.

5.2.2. ADAPTIVE SOLUTION APPROACHES

Adaptive solution approaches aim to optimize a policy: "a rule (or function) that determines a feasible decision given the available information in state S_t" (Powell, 2019). In Powell's definition, a policy can take any shape, including a look-up table, analytical function, or decision tree.

Adaptive solution approaches are applied to both strategic and operational decision contexts. For long-term strategic planning, traditionally, one would optimize for the most likely scenario or an ensemble of scenarios and re-optimize if significant changes occur (W. E. Walker et al., 2001). In contrast, Dynamic Adaptive Policy Pathways is a widely applied approach for adaptive planning under uncertainty (Haasnoot et al., 2013; W. Walker et al., 2013), that finds sequences of actions that maximize the performance of a system over a set of scenarios. The pathways additionally provide insights into lock-ins and path dependencies. Tipping points indicate when to switch strategies.

For operational decision-making, many approaches to adaptive decision-making are rooted in the control literature. Approaches like stochastic dynamic programming (Bellman, 1966) and multi-stage optimization (Bakker et al., 2020; Dupačová et al., 2000) optimize a sequence of actions once, assuming to know the likelihood that scenarios occur. However, we cannot estimate the probability of the occurrence of the various fugitive routes because we have incomplete knowledge of the decision-making of fugitives, and there is inherent variability in human behavior. Another approach, Model Predictive Control, re-optimizes for a rolling planning horizon, considering feedback from the system at each time step (Clarke et al., 1987; Cutler & Ramaker, 1980). However, MPC's typical rolling time horizon is developed for continuous optimization problems and unsuitable for problems with a fixed end time, such as fugitive interception. Periodic Re-Optimization is a variant of MPC that re-optimizes at fixed time intervals until a fixed end time (Psaraftis, 1980). Direct Policy Search (Quinn et al., 2017; Rosenstein & Barto, 2001) and Policy Tree Optimization (Herman & Giuliani, 2018), originating from robotics

	Operational/ control context	Strategic/ long-term context
Re-optimize	*Model Predictive Control; Periodic Re-Optimization*	*Traditional planning approaches*
Pre-optimize	*Direct Policy Search; Policy Tree Optimization*	*Dynamic Adaptive Policy Pathways*

Figure 5.1: Non-exhaustive structured overview of adaptive approaches

and water resource management, pre-optimize for an ensemble of scenarios and determine tipping points at which to change the strategy. These seem to be promising approaches for adaptive decision-making for fugitive interception.

This paper compares Direct Policy Search and Policy Tree Optimization for the fugitive interception problem to evaluate their feasibility for fugitive interception and for time-constrained optimization. The solution quality of Direct Policy Search and Policy Tree Optimization is compared to a Periodic Re-Optimization method to evaluate the value of pre-optimization compared to re-optimization. We choose Periodic Re-Optimization (with a fixed time horizon) over Model Predictive Control (with a rolling time horizon) to ensure consistency and a fair comparison with the other approaches. To benchmark the methods, we assess the solution quality against one-shot optimization, where the strategy remains the same throughout the time horizon, i.e., without utilizing incoming information.

The following paragraphs outline the related literature for the four solution approaches.

ONE-SHOT OPTIMIZATION

One-shot optimization is the benchmark that the online solution approaches are compared to. The strategy remains the same throughout the time horizon, i.e., without utilizing incoming information. Section 5.3.1 describes the formalization of the optimization problem and Section 5.3.2 describes the implementation of one-shot optimization for fugitive interception used in this paper.

PERIODIC RE-OPTIMIZATION (PRO)

A logical first step to utilize incoming information is periodically rerunning the optimization problem. This way, outdated strategies are adjusted to reflect the new situation. This approach is nothing new: there is a long-standing tradition in online optimization, dynamic optimization, and control methods that are built on this principle. Periodic Re-Optimization (Psaraftis, 1980) is a form of closed-loop control, meaning that the actions depend on system output, reflecting how well previous actions have worked. It is an approach to solve dynamic optimization problems, meaning the problem changes over time. A static optimization problem is solved at fixed time intervals - also termed decision epochs or time slices in the dynamic programming literature (Pillac et al., 2013). However, the solutions found by re-optimization do not consider lock-ins, and the computation time of the static optimization problem determines how quickly decision-makers can react to information. Section 5.3.2 describes the implementation of PRO for fugitive interception used in this paper.

DIRECT POLICY SEARCH (DPS)

Introduced by Rosenstein and Barto (2001) for robotics, Direct Policy Search (DPS) is a simulation-based reinforcement learning approach to optimize the parametrization of control policies. The integration of DPS with multi-objective evolutionary algorithms by Giuliani et al. (2016) (Evolutionary Multi-Objective Direct Policy Search (EMODPS)) sparked subsequent applications in water resource management (Gold et al., 2022; Quinn et al., 2017; Zatarain Salazar et al., 2016) and environmental modeling (Marangoni et al., 2021; Rodríguez-Flores et al., 2023).

DPS maps system states to sequential control actions to maximize the expected performance over a specified time horizon. Instead of optimizing periodically to find the best action at each time step, DPS optimizes the parameters θ that describe a control policy p_θ *a priori*. When deployed, this control policy is periodically evaluated with the current system state to obtain the control action at that time. Figure 5.2 shows a schematic overview of DPS. The following paragraphs describe each component in the schematic in more detail.

The simulation model, or transition function $\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{a}_t, \boldsymbol{\varepsilon}_{t+1})$, describes the state of the system at the next time step$\mathbf{x}_{t+1}$, determined by the state of the system at the current time step ($\mathbf{x}_t$), the control actions ($\mathbf{a}_t$), and the external stochastic drivers (ϵ_{t+1}).

At each time step, universal approximators map policy parameters (θ) and system states ($\mathbf{x}_t$) to a vector of control actions ($\mathbf{a}_t$). Universal approximators are, for example, Artificial Neural Networks (ANNs) or Radial Basis Functions (RBFs). Giuliani et al. (2016) compare RBFs to ANNs and demonstrate that RBFs outperform ANNs in their reservoir management case study. They note that the performance of DPS strongly depends on the chosen approximator, and the suitability of approximators depends on the problem characteristics. For example, Oliveira and Loucks (1997) use piecewise linear approximators, while Giuliani et al. (2016) use squared exponential RBFs and Quinn et al. (2017) use cubic RBFs. Zatarain Salazar et al. (2023) systematically compare various Radial Basis Functions and show that the choice of RBFs crucially affects tradeoffs, especially for complex problems. Understanding the structure of the policy improves the choice for the shape of the RBF.

The performance of a policy is given by the objective function J_{p_θ}, which may comprise one or multiple objectives. To account for inherent system variability and uncertainty, the policy is assessed across a set of scenarios or realizations described by ϵ. The objective function (eq. 5.1) aggregates outcomes of the simulation model across realizations and the time horizon, providing a robust measure of the policy's performance.

$$p^*_\theta = \arg\min_{p_\theta} J_{p_\theta} \quad \text{s.t. } \theta \in \Theta; \quad \mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{a}_t, \boldsymbol{\varepsilon}_{t+1}) \tag{5.1}$$

The optimizer suggests values for the RBF parameters that optimize the objective value(s). Zatarain Salazar et al. (2016) and Gupta et al. (2020) evaluate the

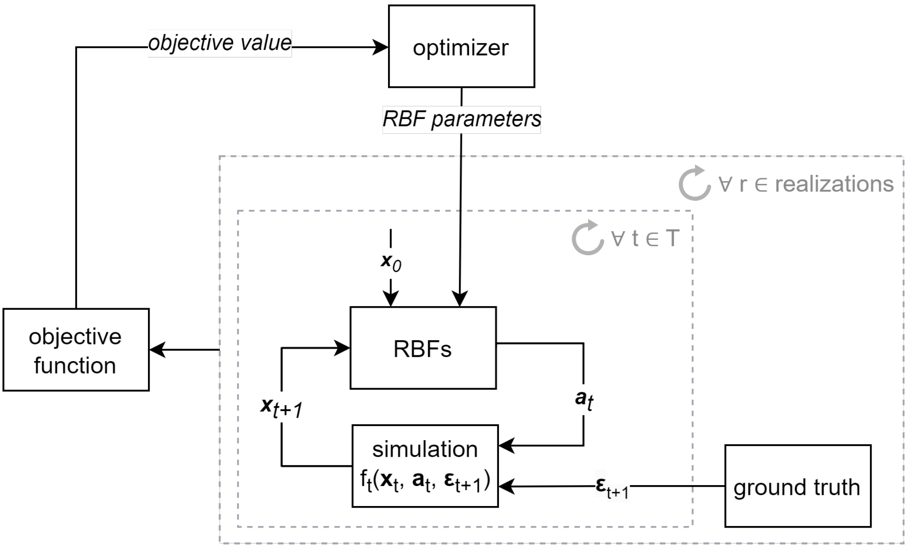


Figure 5.2: Schematic showing the workflow of DPS. The optimizer generates candidate policies for evaluation. Each time step t, given the system state $\mathbf{x}_t$, RBFs translate these into actions a_t. The performance of the policy is aggregated over time steps and realizations. The optimizer uses the objective values to generate new candidate policies.

effectiveness of various Multi-Objective Evolutionary Algorithms (MOEA) for DPS and demonstrate that the Borg MOEA (Hadka & Reed, 2013) outperforms or meets the performance of other MOEAs across test problems.

Section 5.3.2 describes the implementation of DPS for fugitive interception used in this paper.

POLICY TREE OPTIMIZATION (PTO)

Policy Tree Optimization is a simulation-based optimization method where the structure of a binary decision tree is optimized (Herman & Giuliani, 2018). The approach originates in water resource management and has since been applied to several case studies (Cohen & Herman, 2021; Goharian et al., 2022).

An example of a policy tree is given in Figure 5.3. The structure and parameters of the binary tree is optimized using an evolutionary process. First, a population of random trees is generated. Then, the performance of the trees is assessed by evaluating the tree at each time step and noting the simulation output. Based on the relative performance, crossover (the recombination of trees), mutation (local search on the threshold values k_i and actions a_i for all indicator nodes and action nodes), and pruning (the deletion of superfluous branches) generate a new generation of trees. The optimization continues until the maximum number of tree evaluations has been reached.

Section 5.3.2 describes the implementation of PTO for fugitive interception used in this paper.

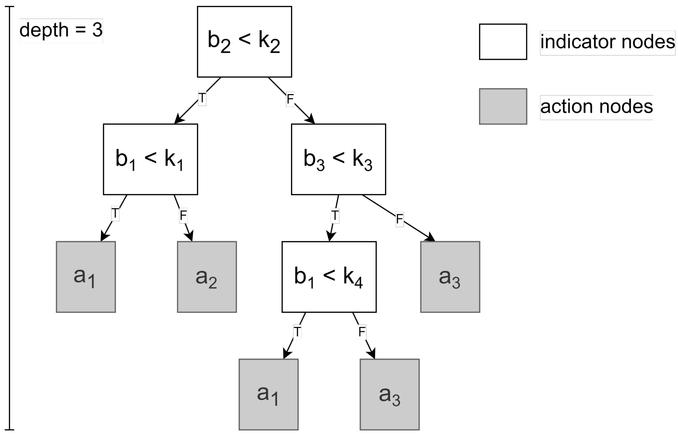


Figure 5.3: A policy tree of depth 3, with indicator nodes in white with indicator variables b_t and thresholds k, and action nodes with actions a_i in gray (based on (Herman & Giuliani, 2018)).

5

5.3. METHOD

5.3.1. OPTIMIZATION PROBLEM

We model the problem using the sequential simulation-optimization approach as presented in Van Droffelaar et al. (2024b), where a simulation model is repeatedly run to generate a set of scenarios, which are subsequently used as input for an optimization model (Figueira & Almada-Lobo, 2014). Specifically, the behavior of the fugitive is described by a Discrete Event Simulation model, and the target positions of the police units are subsequently optimized. In this paper, we substitute the optimization model with each of the considered adaptive approaches. The following paragraphs describe the optimization model of the intercepting police units, the simulation model of the fugitive, and the information updates.

FORMALIZATION OF THE OPTIMIZATION PROBLEM

The optimization model is adapted from the Flow Interception Problem, introduced by Hodgson (1990) and Berman et al. (1992). The decision variables of the optimization problem are the police unit positions $\pi_{u,v}$ and the intercepted routes z_r (Table 5.1). Equations 5.2-5.4 describe the generic formalization of the optimization problem. Essentially, a route is intercepted ($z_r = 1$) if, for that particular route (r), the fugitive is at the same place (v) at the same time (t) as the target vertex of a police unit ($\pi_{u,v}$), and that target vertex is within reach at that time for that particular police unit u ($\tau_{u,v,t}$). The target vertices of the police units are optimized to maximize the number of intercepted escape routes. Routes are only intercepted at a target vertex - not at intermediate intersections. The *min* function encodes that each route can only be intercepted – and count towards the objective function – once. This objective function is implemented in each of the solution approaches used in this paper.

Table 5.1: Notation of parameters and decision variables.

Decision variables	
$z_r \in \{0,1\}$	route r is intercepted
$\pi_u \in V$	target vertex of police unit u
$\pi_{u,v} \in \{0,1\}$	vertex v is the target vertex of police unit u
Parameters	
$V = \{v\}$	set of vertices
$R = \{r\}$	set of fugitive routes
$U = \{u\}$	set of police units
$S = \{s\}$	set of sensors
$T = \{t\}$	ordered index set of time steps
t_{max}	maximum time step; length of planning horizon
$\phi_{r,v,t} = \{0,1\}$	fugitive route r is present at vertex v at time step t
$\tau_{u,v,t} = \{0,1\}$	vertex v is reachable by police unit u at time t

$$\text{Maximize: } Z = \sum_{r \in R} z_r \tag{5.2}$$

$$\text{Subject to: } \sum_{v \in V} \pi_{u,v} = 1 \qquad \forall u \in U \tag{5.3}$$

$$z_r = \min\Big(1, \sum_{u \in U} \sum_{t \in T} \sum_{v \in V} \phi_{r,v,t} \cdot \pi_{u,v} \cdot \tau_{u,v,t}\Big) \qquad \forall r \in R \tag{5.4}$$

SIMULATION OF THE FUGITIVE ESCAPE ROUTES

We model the route choices of the fugitive as the shortest paths from the incident location to each of the escape vertices of the network and generate a set of plausible escape routes by running this model with different random seeds. To obtain a diverse set of routes, 2% noise is added to the routes, meaning that the suspect takes a wrong turn every 2% of the intersections (van Droffelaar et al., 2024a). After a wrong turn, the new shortest path is determined from their next position. This noise accounts for three factors: (1) simulating human error, especially in stressful situations; (2) accommodating adjustments for unexpected obstacles like red traffic lights; and (3) accounting for factors not explicitly modeled. As a result, we observe a distribution of routes around the optimal paths. In further research, this model can, of course, be replaced by a richer behavioral model.

INFORMATION UPDATES

Besides, there are sensors on the networks that represent the traffic cameras with automatic number plate recognition that the police can access. These sensors do not intercept the fugitive routes but provide information about the fugitive's location. Upon detection, the set of fugitive routes collapses to a narrower set of possible routes. At each time step t, all sensor detections from the preceding time interval $[t-1, t]$ are considered to adjust the strategy. We employ a ground truth approach to generate the sensor detections, meaning one fugitive route (the ground truth) generates the considered input. If the ground truth fugitive route r passes a sensor s in the time interval $[t-1, t]$, the sensor detection is 1 for sensor s at time t. For each repetition, another fugitive route in R is the ground truth route. Specific descriptions of the implementation of sensor detections for each solution approach are included in the next section.

5.3.2. SOLUTION APPROACHES

This paper compares the adaptive solution approaches Periodic Re-Optimization, Direct Policy Search and Policy Tree optimization to a static one-shot optimization. The following sections describe the implementation of each approach for fugitive interception.

ONE-SHOT OPTIMIZATION

5

The one-shot optimization model maximizes the fraction of the generated set of plausible escape routes that are intercepted by optimizing the positions of available police units, as described in Section 5.3.1. An escape route is intercepted when it occupies the same vertex at the same time as a police unit. The model is optimized using a metaheuristic solution approach for 10 000 function evaluations. The optimization runs once and is *not* updated, regardless of sensor detections. We use a basic genetic algorithm complemented by Borg's auto-adaptive framework, which co-evolves evolutionary operators' probabilities for population adaptation (Hadka & Reed, 2013). The operators, each initially assigned equal probability during the algorithm's initialization, contribute to population adaptation based on their relative success. The operators used are Simulated Binary Crossover, Differential Evolution, Parent-Centric Crossover, Simplex Crossover, Unimodal Normal Distribution Crossover, and Uniform Mutation. We use the default values for all hyperparameters as presented by Hadka and Reed (2013).

PERIODIC RE-OPTIMIZATION (PRO)

Periodic re-optimization solves a static optimization problem at fixed time intervals. Based on the information updates, the set of escape routes is updated to exclude routes that are no longer possible given the received sensor detections. To generate the information inputs to the optimization model, we pick a ground truth escape route from the set of generated escape routes for each repetition. The ground truth route triggers sensor detections. The generated escape routes are filtered at each time step based on their feasibility given the received sensor detections. Then, the one-shot optimization is run based on this ever-shrinking subset of routes. We record for each repetition whether or not the re-optimization has led to interception, and we take the average over the set of ground truth escape routes to obtain the probability of interception.

The optimization is run for 10 000 function evaluations for 10 seeds each time the set of possible routes changes due to sensor detections (or lack thereof). The number of re-optimizations is dependent on the number of sensors and the problem instance. For problem instances with three sensors, this change happens on average 2.2 times for the grid network, 3.7 for the Manhattan network, 4.0 times for the Utrecht network, and 3.1 times for the Winterswijk network. For problem instances with three sensors, this change happens on average 3.2 times for the grid network, 5.3 for the Manhattan network, 6.5 times for the Utrecht network, and 5.2 times for the Winterswijk network. Consequently, the total number of function evaluations varies between 22 000 and 65 000 function evaluations. In practice, however, police vehicles can already start driving towards their first position after the first 10 000 function evaluations. In contrast, with pre-optimization approaches like DPS and PTO, they have to wait until the entire optimization is complete before starting to drive. Therefore, for the main experiments, the optimization is run for 10 000 function evaluations. Additional analyses in C.2.1 explore the impact of capping the *total* number of function evaluations to 10 000.

DIRECT POLICY SEARCH (DPS)

Direct Policy Search optimizes the parametrization of control policies that map system states to sequential control actions. This mapping has the shape of a Radial Basis Function (RBF). A transition function describes the progression of the system state, depending on the control actions resulting from the optimized control policy.

For fugitive interception, the transition function details how the positions of the police units and the fugitive change given the chosen actions and the ground truth route. The ground truth route generates sensor detections, which make up the state vectors $\mathbf{x}_t$. For improved convergence, the sensor stays flipped after a detection (i.e., the vector looks like [000111], rather than [000100] for a detection at time step 3).

For each time step, the control policy is evaluated for each police unit to obtain its target vertex. Experimentation showed that representing the control policy by a linear RBF with n=2 leads to the quickest convergence for this problem, compared to a Gaussian or cubic RBF with n=2 or n=6 (Equation 5.5). $c_{s,j}$, $r_{s,j}$, and $w_{u,j}$ are the centers, radii, and weights of n linear RBFs. The weights are specific to each police unit (i.e., the decision variables), and the centers and radii are specific to the sensors (i.e., the information). The resulting actions $a_{u,t}$ are clipped to fall within the bounds of the decision variable.

$$a_{u,t} = \sum_{s \in S} \sum_{j=1}^{n} w_{u,j} \left(c_{s,j} \cdot x_{s,t} + r_{s,j} \right) \qquad \forall u \in U, \forall t \in T \tag{5.5}$$

Analogous to other solution approaches, the algorithm is run for 10 000 function evaluations.

POLICY TREE OPTIMIZATION (PTO)

Policy Tree Optimization optimizes the structure of a binary tree, where the sensor information is on the indicator nodes (has sensor A detected the fugitive or not), and the positions of the police units are on the action nodes. We optimize a policy tree for each police unit to ensure that each police unit can receive a new position at each time step. We adopt the evolutionary scheme implemented by Herman and Giuliani (2018) but supplement the evaluation of the trees with a robustness consideration. Each group of trees is evaluated for 100 ground truth realizations of the fugitive route, and we record the number of realizations in which the route is successfully intercepted. Analogous to PRO and DPS, the ground truth fugitive route generates the sensor detections that serve as input to the policy tree. Like DPS, the sensor stays flipped after detection to improve convergence (i.e., the vector looks like [000111], rather than [000100] for detection at time step 3). Analogous to other solution approaches, the algorithm is run for 10 000 function evaluations.

5.3.3. EXPERIMENTAL SETUP

CASE STUDY ROAD NETWORKS

We evaluate the solution approaches on three types of networks to gently build complexity and assess the generalizability of our conclusions to different graph topologies.

1. A demonstration network to observe how the various approaches treat incoming information.

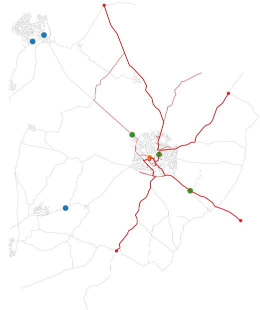
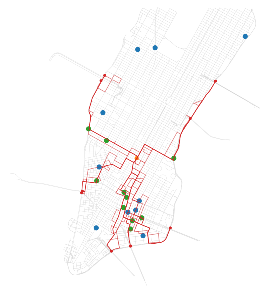
2. A 2D 10x10 grid with equidistant vertices, with travel time over each link equal to the time step. The starting positions are in the center of the grid, and the designated escape vertices are on the boundary of the grid.

3. Three city road networks to assess the generalizability to real-world decision contexts. The three networks are selected for their different topologies: Utrecht, the Netherlands, represents a typical European city with a historical center surrounded by modern neighborhoods; Winterswijk represents a rural area with sparse roads surrounding a town; and Manhattan represents a modern city with a grid layout. The starting positions are in the center of the network, and the escape vertices are the boundaries of the city (see also Table 4.2).

PROBLEM INSTANCES

Besides the network topology, the starting positions of the fugitive and the police units determine the complexity of the optimization problem: the more overlap between vertices visited by the fugitive and vertices reachable by police units, the more potential interception positions. The positions of the sensors determine whether, to what extent, and how often strategies should switch. Therefore, we control for the influence of these factors on our results by experimenting with 10 different problem instances. Each problem instance describes a different configuration of the problem: a starting position for the fugitive, starting positions for the police units, and positions of the sensors. A set of fugitive routes is generated for each starting position of the fugitive. The sampled positions are checked for three criteria and resampled if necessary: (1) the starting position of the fugitive does not overlap with the starting position of a police unit; (2) each sensor is passed by

Table 5.2: Case study road networks used in this study. Escape vertices and simulated escape routes are marked in red. One set of starting positions is displayed in blue (police units) and green (sensors).

Utrecht	Winterswijk	Manhattan
Typical European city Incident: city center Escape: reach the highway Size: 6001 vertices	Rural area Incident: city center Escape: cross the border Size: 4892 vertices	Modern, grid city Incident: Union Square Escape: get off the peninsula Size: 6419 vertices
		

at least one fugitive route; (3) each police units can reach at least one fugitive route within the run length of the simulation. This last constraint ensures that all police units can contribute to the interception, which contributes to a fair comparison between problem instances. The solution approaches are evaluated on the same problem instances to ensure a fair comparison.

Metrics for comparison

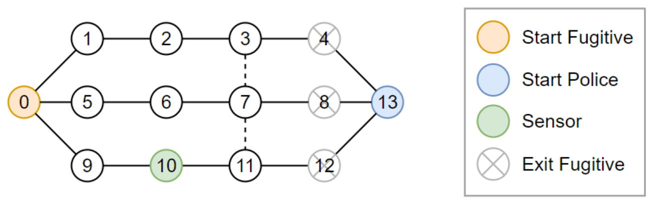
The police have, at most, a few minutes to determine where to send available police units (Mehlbaum et al., 2014). Therefore, timely calculation of the interception positions is essential to support police interception operations. Therefore, we compare the performance of the solution approaches in this paper based on the quality of the solution found after 10 000 function evaluations, rather than based on their best attainable solution when converged. For each problem instance, only consider the best-obtained solution across 10 random seeds for the optimization algorithm. The number of function evaluations is a proxy for the computation time. Due to varying degrees of code optimization and parallelization, comparing the solution quality after a set computation time would not be fair. We discuss this in more detail in the discussion (Section 5.5).

5.4. Results

The following subsections present and discuss the results for the various networks: firstly, a test network; second, a ten-by-ten grid; and last, three city road networks to evaluate the influence of network topology and assess the generalizability of the results.

5.4.1. TEST NETWORK

First, we consider a network (Figure 5.4) where a police unit and a fugitive start on opposite ends and three paths of equal length connect their respective vertices. The target vertices of the fugitive are vertices 4, 8 and 12, and the fugitive is considered to be 'escaped' when they reach one of these vertices. To reach these vertices, there are three possible routes for the fugitive: through vertices 1, 5, or 8. The police have additional paths between vertices 3 and 7, and 7 and 11, that allow them to switch between paths. The fugitive cannot use these.



5

Figure 5.4: Network layout of the test problem

We optimize the target vertices of the police unit to maximize the probability of interception. The fugitive is considered 'intercepted' when they are at the same vertex at the same time as the police unit.

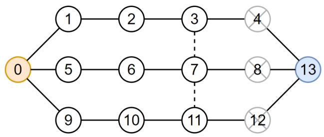
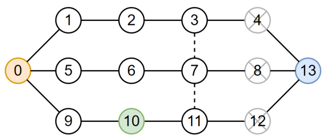
If there is no sensor, i.e., no additional information about the location of the fugitive becomes available during the interception attempt, and the paths are equally attractive to the fugitive, the probability of intercepting the fugitive is 33%. The one-shot approach, periodic re-optimization, direct policy search, and policy tree optimization all find this solution (Table 5.3).

A sensor on vertex 10 increases the probability of interception for all reactive approaches. The pre-optimized adaptive approaches — PTO and DPS — both quickly (in less than 2 generations) find the optimal solution where 67% of routes are intercepted (Table 5.3). This is the solution where the police unit is either positioned on vertex 7 or remains on vertex 13 and goes to vertex 11 or 12 if the sensor detects the fugitive and goes to vertex 7 or 8 if the sensor does not detect the fugitive. The escape routes on the top path, escaping from vertex 4, are not intercepted. Periodic re-optimization finds a slightly worse solution because the first calculated position is on the top path in some repetitions. If the sensor then detects the fugitive on the bottom path, the police unit does not have enough time to reach them. This demonstrates the path dependency in online optimization of fugitive interception.

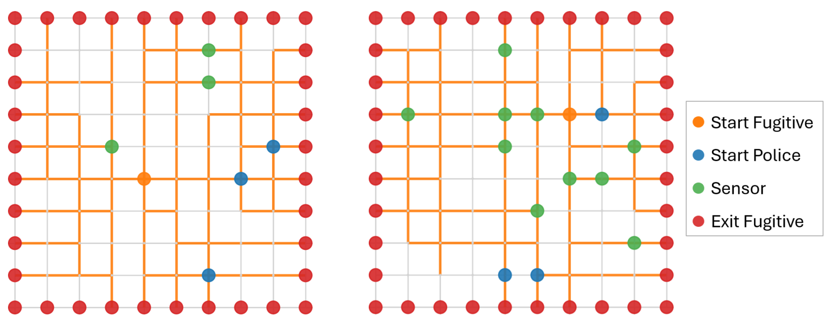
5.4.2. 2D GRID NETWORK

Second, we consider a ten-by-ten undirected grid network with equidistant vertices, where the fugitive starts in one of the nine middle vertices and can escape at any of the 36 edge vertices. The police starting positions and sensor locations are randomly positioned on the network, leading to ten different problem instances

Table 5.3: Fraction of intercepted routes for each solution approach for the test network.

	one-shot	PRO	DPS	PTO
	*0.33*	*0.33*	*0.33 (100 nfe avg)*	*0.33 (100 nfe avg)*
	*0.33*	*0.58*	*0.67 (175 nfe avg)*	*0.67 (135 nfe avg)*

(illustrative examples in Figure 5.5). We examine situations with three and ten police units ($|U|$) and with three and ten sensors ($|S|$). The results in Figure 5.6 are scaled to the best solution across seeds and across approaches. Each dot represents the best solution for a problem instance across seeds. The overlaid boxes represent the first quartile, median, and third quartile of the scaled solution quality, respectively.



(a) Problem instance with three police units and three sensors.

(b) Problem instance with three police units and ten sensors.

Figure 5.5: Example problem instances for the ten-by-ten grid network. Fugitive starting locations and simulated escape routes are marked in orange, and escape vertices are red. Combinations of starting positions are displayed in blue (police units) and green (sensors).

The results (Figure 5.6) show that, in general, adaptive approaches outperform one-shot optimization in terms of solution quality. With an increasing number of sensors, the (unscaled) solution quality increases (Figure C.1), demonstrating that additional information on the whereabouts of the fugitive improves the probability of interception.

One problem instance with three police units demonstrates PRO's vulnerability to path dependence. For this particular instance, the difference between the pre-optimized (DPS and PTO) and re-optimized (PRO) approaches is significant: PTO

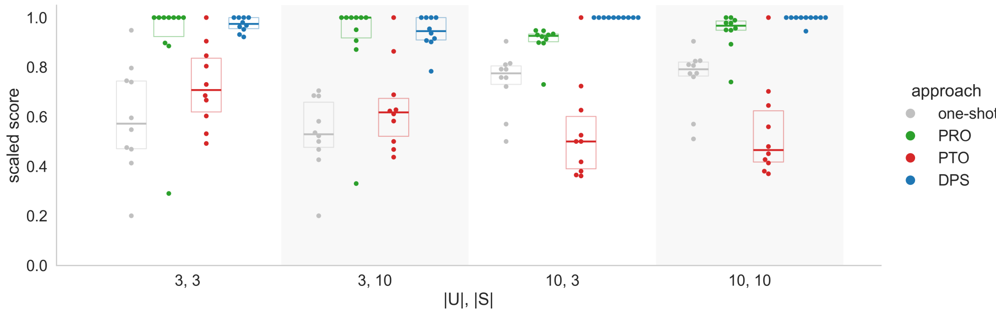


Figure 5.6: Comparison of solution quality on a 10x10 grid across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

5

and DPS find a solution with a 100% probability of interception, whereas one-shot optimization and PRO get stuck at 23% and 29%, respectively.

Across problem instances, PTO cannot find the best solution within the given number of function evaluations. This is particularly visible in problem instances with ten police units, where even one-shot optimization outperforms PTO in solution quality. Policy trees are not as expressive as PRO and DPS and suffer from a difficult evolution process. Crossovers, where pieces of policy trees are exchanged between solutions, rarely produce feasible, well-performing policy trees. Therefore, PTO quickly converges to and rarely escapes from a local optimum, making it sensitive to the initial sample. DPS and PRO yield similar solution quality. PRO performs slightly better for problem instances with three police units, and DPS performs slightly better for instances with ten police units.

5.4.3. CITY ROAD NETWORKS

MANHATTAN

The Manhattan road network is similar to the grid discussed in the previous subsection, albeit with some one-way roads and other irregularities and fewer escape vertices (see Table 4.2). However, the results (Figure 5.7) show different trends compared to the results of the grid network. The spread in solution quality across problem instances is much lower for all solution approaches. In other words, the starting position of the police units and sensor locations has a smaller impact on the ability of the various solution approaches to find a good solution. Evidently, the underlying road network and simulation of escape routes affect the relative performance of solution approaches.

Compared to the results for the grid network, one-shot optimization and policy tree optimization perform much better. This is most likely due to the more limited set of unique routes and, therefore, the presence of clear bottleneck interception positions. Ten police units are enough to cover all major bottlenecks, leading to very good performance for one-shot optimization. With three police units to be positioned, PRO performs slightly better than other approaches for problem in-

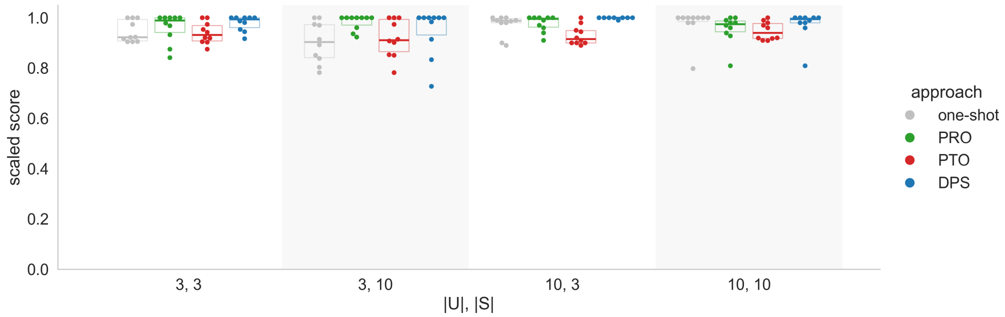


Figure 5.7: Comparison of solution quality for the Manhattan road network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

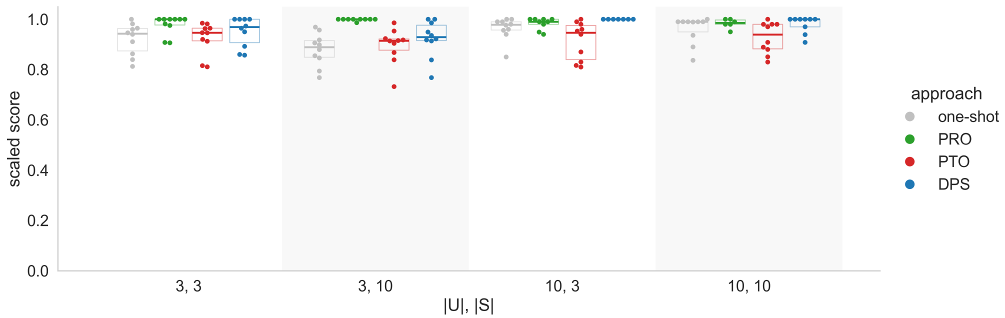


Figure 5.8: Comparison of solution quality on the Utrecht road network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

stances with three police units, and DPS performs best for instances with ten police units. This is consistent with the results for the grid network.

UTRECHT

Compared to Manhattan, the road network of Utrecht is more complicated, with one-way roads and cul-de-sacs, and hierarchical, with neighborhoods with main roads connecting different parts of the city. Despite the differing network topology, the results (Figure 5.8) are similar: PRO performs best on problem instances with three police units, and DPS performs best on problem instances with ten police units. For problem instances where PTO and DPS show a lower solution quality, this is due to the low number of function evaluations, which does not allow for convergence.

WINTERSWIJK

The road network of Winterswijk consists of a town surrounded by several larger roads that lead to the German border, which forms the exit vertices. The results (Figure 5.9, C.4) show that deploying ten police units is sufficient to guarantee in-

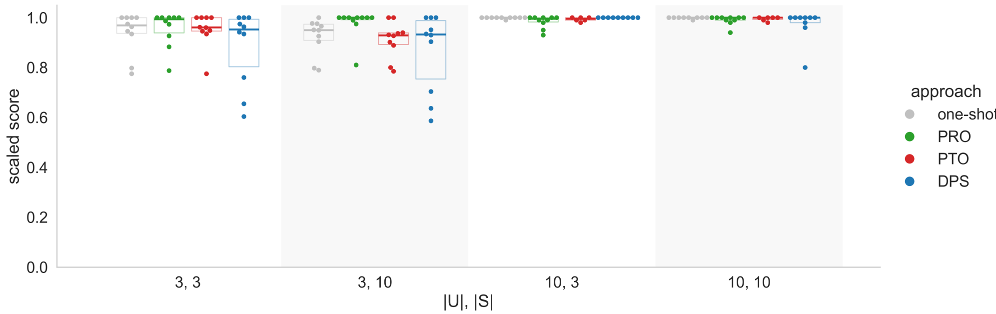


Figure 5.9: Comparison of solution quality on the Winterswijk road network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

terception of all simulated escape routes. In other words, utilizing information to improve the solution quality does not improve the probability of interception. On the contrary, adding adaptation slows convergence, and therefore, the best solution may not be found.

Similar to the results for Utrecht and Manhattan, PRO performs best on problem instances with three police units. In contrast to the previous networks, PTO performs better than DPS, which struggles to find a good solution for three problem instances.

5.4.4. CONVERGENCE

Additional analyses were performed to understand the underlying mechanisms leading to the results discussed in this section. The number of function evaluations is increased or decreased to evaluate the effect of convergence on the results.

PRO gets 10 000 function evaluations for each re-optimization, which occurs 3 - 7 times on average, depending on the problem instance, resulting in 30 000 - 70 000 function evaluations in total. C.2.1 presents the results limiting the number of function evaluations of PRO to 10 000 in total (instead of per re-optimization). Problem instances with many re-optimizations (Utrecht), more police units, and more nodes visited by the simulated escape routes — and therefore more potential interception positions — (Manhattan and Utrecht) suffer significantly from limiting the number of function evaluations.

For a few problem instances where PTO performs relatively poorly, additional experiments with 100 000 function evaluations are run (C.2.3). Across networks and problem instances, the quality of the solution gets better, with solutions closer to the best-found solution, indicating that slow convergence is an important factor for the lower solution quality.

Similarly, for a few problem instances where DPS performs relatively poorly, additional experiments with 100 000 function evaluations are run (C.2.2). The quality of the solution rarely improves after 10 000 nfe, demonstrating that the performance of DPS is not solely hindered by the number of function evaluations.

5.5. DISCUSSION

5.5.1. REFLECTIONS ON SOLUTION APPROACHES

In this paper, we compare the performance of one-shot optimization, periodic re-optimization, direct policy search, and policy tree optimization for real-time fugitive interception. First, we discuss the solution quality obtained by each solution approach.

- One-shot optimization is a non-reactive solution approach that functions as a benchmark in this paper. This representation of the optimization problem is easier to solve, and the optimization is at or close to convergence at 10 000 nfe. For problem instances with ten police units, one-shot optimization often finds the best-found solution quality, showing that using information is more critical for problem instances with fewer police units.

- Periodic re-optimization reruns the one-shot optimization problem each time the sensor information excludes potential escape routes. This approach comparatively gets more function evaluations because not everything needs to be computed before being able to send out police units, and therefore, it performs very well. Experiments where PRO gets 10 000 nfe *in total* show PRO lacking behind DPS. Some problem instances demonstrate PRO's vulnerability to path dependence. This is less common in the city network used in this paper because the complexity of the road network allows for quick pivoting. In a highway network, where pivoting is much more time-consuming, PRO may be more vulnerable to lock-ins, similar to the test graph discussed in Section 5.4.1.

- Direct policy search performs well across networks and problem instances, especially considering the low number of function evaluations. The solution quality obtained by DPS depends strongly on the initial sample, and extending the optimization to 100 000 function evaluations barely improves the solution quality further. This suggests that improving the algorithm by including restarts or a better-informed initial sample will likely improve the solution quality found by DPS (Fukunaga, 1998; Hadka & Reed, 2013).

- Policy tree optimization is outperformed by PRO and DPS across networks and problem instances. Experiments with 100 000 function evaluations demonstrate that PTO often can find a good solution but converges too slowly for real-time decision support. The evolutionary process leads to many infeasible solutions and slows convergence. Tailoring the evolutionary processes of crossover and mutation to the fugitive interception problem will likely improve the real-time performance of PTO. Additionally, including restarts would prevent PTO from getting stuck in local optimums, and an informed initial sample would further speed up convergence.

5.5.2. INTERPRETABILITY OF SOLUTIONS

Adaptive solution approaches not only improve solution quality but also improve the interpretability of the interception strategy. In contrast, a one-shot optimization might position units in one area even if they know the fugitive has been detected on the other side of the city. Additionally, pre-computed solution approaches (DPS and PTO) lead to fewer changes in interception positions, and the changes can be anticipated and understood by individual police units. This transparency could lead to higher trust and acceptance of the decision support system (Shibl et al., 2013). Previous research on decision support systems for the Dutch Police found that it is crucial for compliance that individual police agents understand the reasoning behind the decision support and their individual contribution to the interception (Drenth & Steden, 2017). Adaptive solution approaches contribute to this.

Interpretability (the ability to explain or to provide the meaning of a model in understandable terms to a human, as understood by Barredo Arrieta et al. (2020)) builds trust and acceptance of the decision support. Rudin (2019) even argues that 'black box models' (models where the parameters and architecture are hidden) are insufficient in high-stakes decision environments and should be avoided if possible. The most interpretable solution approach is one-shot optimization, which presents a single combination of interception positions after an incident. While the output of each re-optimization of PRO is interpretable, the solution approach is not inherently transparent about what information change triggered the re-optimization. Instead, policy trees are designed to be an easily interpretable structure that visualizes the conditions triggering various actions. In contrast, direct policy search yields an incomprehensible combination of fitted radial basis functions that map system states to interception positions. Further research could develop an approach to map the functions to a binary decision tree, combining the convergence speed of DPS and the interpretability of PTO.

5.5.3. FURTHER RESEARCH

In this paper, we make a number of assumptions and simplifications in the representation of fugitive interception in our models. In the following paragraphs, we discuss the assumptions and their impact on the various solution approaches.

First, this paper assumes that interception is deterministic: when a police unit and the fugitive are at the same intersection at the same time, the fugitive is always intercepted. However, in reality, busy intersections or multiple highway lanes are difficult to monitor, and a fugitive can slip through unseen. Police interception strategies hedge for this by assigning multiple police units to important bottlenecks. Simulation-optimization for decision support could take this into account by assuming probabilistic interception in the objective function. For example, only 80% of a fugitive route is intercepted by each intercepting police unit. This percentage could also depend on the interception position's characteristics, such as the number of lanes or typical traffic density. The implementation of probabilistic interception is similarly simple for all solution approaches discussed in this paper.

Second, this paper assumes accurate detection of the fugitive. However, in reality, Automatic Number Plate Recognition (ANPR) Cameras may produce false positives and false negatives. Accounting for these inaccuracies is very difficult in pre-computed adaptive approaches that rely on threshold values to determine the interception positions at each time interval. For periodic re-optimization, implementing inaccurate detection would simply imply changing the reward of intercepting a route (z_r) from a binary variable ($z_r \in \{0,1\}$) to probabilistic values, such as $\{0.05, 1\}$.

Third, this paper only considers detection by ANPR cameras, which are located at fixed, known points in the network. However, civilian reports of abnormal behavior (e.g., driving on the wrong side of the road or at excessive speeds) could also inform police interception. In such cases, the detection location is not fixed and can occur anywhere on the road network at any time. Pre-computed adaptive approaches cannot account for this, as the state space would explode in size, making real-time calculation of interception strategies infeasible. For periodic re-optimization, such detections would either change the relative rewards of different simulated escape routes or simply exclude routes for the next re-optimization.

Finally, the experiments in this paper assume zero computation time, meaning there is no delay between the start of the fugitive's escape and the police units' response. However, computation time varies based on factors such as hardware, programming language, and code efficiency, making fair comparisons challenging. Further research should examine the relationship between the number of function evaluations (which influences the computation time), the solution quality (which depends on the number of function evaluations), and the delay in police response (which is tied to computation time). For the solution approaches discussed in this paper, DPS and PTO initially have longer computation times, but they determine the appropriate actions within milliseconds during the interception. On the other hand, the initial computation time of PRO is shorter, but this computation time is required at every re-optimization. While different detection scenarios can be pre-calculated at the start of an interception, this is computationally intensive and only possible for ANPR detections at fixed, known locations. In summary, the effect of non-zero computation time on the effectiveness of the solution approaches is not yet known and needs further experimentation.

5.5.4. RECOMMENDATIONS

Adaptive solution approaches increase the probability of interception and are expected to improve trust in the decision support due to their added transparency. Considering the solution quality in limited function evaluations and the interpretability of results — both to intercepting police units and dispatchers in the control room —, DPS is a promising approach for decision support, especially if supplemented with an interpretable interface. However, in city networks with low vulnerability to lock-ins, and if the police want to consider probabilistic interception, probabilistic detection, and non-ANPR detections, PRO is the most flexible and effective option.

5.5.5. GENERALIZABILITY

The interception problem described in this paper is an example of a class of problems where an optimal intervention that reacts to observations of the system has to be determined. Another example of this class of problems is the control of an autonomous vehicle. At fixed, short time intervals, the best control action has to be determined, meaning that the car follows its intended route and avoids crashes while the behavior of the vehicles around it is unknown (Zanon et al., 2014). Besides being a good control action for the current state of the system, it should take into account possible changes in the system, such as unexpected other traffic. Lock-ins that make it difficult to react to changes in the system should be avoided. Moreover, similar to the fugitive interception problem discussed in this paper, the control action must be available quickly — within a second or even less. Model Predictive Controllers for autonomous driving are widely researched Cesari et al. (2017) and Lamouik et al. (2023), but could benefit from pre-computation solution approaches to anticipate path-dependence and prevent lock-ins. Different problems will have different trade-offs: the simpler the optimization problem and the more vulnerable to high-impact lock-ins, the further the preference for a solution approach shifts to DPS. In problem areas where the interpretability of the solution is paramount the preference shifts to PTO.

5.6. CONCLUSION

Simulation–optimization can be used to support real-time decision-making for fugitive interception. Incorporating real-time information about the fugitive's location can improve decision-making, but it also makes the optimization problem harder to solve within the limited time available for real-time decision making.

This paper examines the solution quality obtained for the fugitive interception problem with information updates within a limited number of function evaluations for four solution approaches: One-shot Optimization, Periodic Re-Optimization, Direct Policy Search, and Policy Tree Optimization. Our analysis using the fugitive interception example shows that:

1. Adaptive solution approaches, utilizing the information updates, outperform static one-shot optimization, especially when the number of intercepting police units is low.

2. Direct Policy Search effectively avoids lock-ins and finds high-quality solutions across networks and problem instances. However, making the resulting solutions interpretable for decision-makers requires further research.

3. Policy Tree Optimization, while the most interpretable, converges too slowly for real-time decision support.

4. Periodic Re-Optimization performs well for networks and problem instances with few lock-ins (for instance, city networks) and is flexible to further extensions to include probabilistic interception and detection. While the optimization results at each time step are interpretable, Periodic

> Re-optimization does not offer insight into causal relationships between sensor information updates and changes in calculated positions of police units.

Based on the results in this paper, practitioners are advised to use Direct Policy Search for problems that are vulnerable to lock-ins. If the interpretability of the results is critical, Direct Policy Search should be supplemented with an interpretable interface. Otherwise, practitioners are advised to use Periodic Re-optimization for its flexibility and ease of implementation.

These results are generalizable to a class of problems where an optimal intervention has to be determined in real time under uncertainty, and information updates can improve the intervention, such as path planning of autonomous vehicles.

6

EMPIRICAL EVALUATION

The previous chapters focused on developing models and testing various solution approaches for fugitive interception. This chapter evaluates these models by comparing their outputs to the actual locations where police units were positioned by the control room. This evaluation leads to a discussion of limitations and suggestions for future research.

6.1. INTRODUCTION

To apply models in the police control room to advise on interception strategies, the models need to be thoroughly validated and tested. The evaluation discussed in this chapter is the first step in this process. The evaluation helps to identify strengths and limitations in the proposed approach, and leads to recommendations for improvements and further research.

Validation is the assessment of the accuracy of the model's representation of the real system (Sargent, 2011). In fugitive interception and, more broadly, in law enforcement applications, validation is extremely difficult. Validation of models of fugitive behavior is difficult because there is very little data about fugitive escape routes. When data is available, this data has a historical bias (criminals constantly change their strategies, especially when the police start catching on) and survivorship bias (we only have information on successful cases where the suspect was caught). Furthermore, the data collection in police systems does not facilitate simulation model validation. On the other hand, validation of the police interception positions, for example, to assess whether interception positions were reachable in the estimated time, is possible.

In this chapter, we take an initial step toward model validation by empirically evaluating the simulated escape routes and the calculated police interception positions for three historical cases. For each case, we answer the following questions:

6

1. Is the location of the first sighting of the fleeing suspect included in the simulated escape routes?
2. How effective are the real police positions in intercepting simulated routes?
3. How effective and efficient are the calculated police interception positions in intercepting simulated routes?
4. How do the real and calculated positions differ and why?

Effectiveness refers to the percentage of simulated routes intercepted by the police positions. Efficiency, on the other hand, examines whether the same interception percentage could be achieved using fewer police units. These metrics are further explained in Section 6.2.4.

6.2. METHOD

6.2.1. DATA SOURCE

Control room centralists from the fire department, the ambulance dispatch center and the police use the 'Geïntegreerd Meldkamer Systeem' (GMS) to handle incident reports and direct the right emergency service to the right place as quickly as possible. Each incident in GMS is classified to distinguish, for example, home burglaries or resuscitations. Centralists use the free text fields ('kladblok') to note information relevant to each incident. Each entry to the free text fields is timestamped. Communication with police units in the field is done primarily through radios ('portofoon'), which is not recorded.

For this research, we had access to the free text fields of GMS of incidents labeled 'chase'/ 'achtervolging'. This label includes escapes by car and interceptions as defined in this thesis. To ensure that we do not affect police operations, we only consider cases older than 3 years.

6.2.2. SELECTION OF CASES

To align with the models developed in this dissertation, we applied criteria to the dataset of incidents. First, we filtered all cases by excluding keywords:

- Escape by car (not by foot, bike, or moped). *('te voet'; 'fiets'; 'scooter')*
- The suspect was not under the influence of drugs or alcohol. Behaviors associated with intoxication aren't captured by the models of fugitive behavior in this dissertation. *('dronken'; 'alcohol'; 'onder invloed')*
- No kids were involved. *('kind')*

Additionally, we filtered the dataset using the timestamps:

- The duration of the incident from the first notification to the last communication is at least 15 minutes. This excludes incidents that were not followed up on or are concluded so fast they are not relevant to this dissertation.
- There are at least three entries. This excludes incidents that were not documented well enough to use in this evaluation.

After applying these filters, we screened the remaining incidents and selected cases where:

- The suspects fled using a single car. Cases involving multiple vehicles are excluded.
- The suspect does not switch vehicles.
- The police interception positions are documented well. The dataset includes many instances where no or a fraction of the police interception positions are noted in GMS, which makes evaluation difficult. This is very common, especially for incidents with high time pressure (which applies to most of the incidents).
- It is possible to rationalize a set of plausible target locations for the suspect, without using personal data or confidential police information. For the approach proposed in this dissertation, this is needed to simulate the escape routes.

One type of incident that fits these criteria is ATM burglaries using explosions or a ram-raid ('ram- en plofkraken'). These ATM raids are abundant and impactful causing massive damage to buildings and surroundings (Politie, 20 April 2024). With the improved security of Dutch ATMs, Dutch raid groups are targeting German and Belgian ATMs and driving back to the Netherlands in stolen fast vehicles (van der Eng, 2018). This escape strategy has been well-covered in Dutch and German press (NOS Nieuws, 7 August 2017, 25 July 2017), meaning we can safely use these types of cases in this dissertation. Since the incident location is in Germany, the Dutch police have ample time to develop an interception strategy and position police vehicles accordingly. Generally, the interception positions are relatively well-documented.

From the ATM raids in the dataset, we selected three with high data quality, meaning that the interception positions are recorded, and, if the fleeing suspects were sighted, their location.

6.2.3. Confidentiality

We take several precautions to ensure that the selected cases cannot be traced back to the related incidents and that no confidential information is disclosed. First, no specific details or dates related to the case are shared. Second, the starting positions of the fugitives are not plotted at the real location of the incident, but within the general area. Third, the locations where the fugitives were spotted or arrested are not disclosed.

A security clearance and authorization to work with sensitive police data were obtained for carrying out the data screening and evaluation. The Dutch Ministry of Justice and Security has approved the use of police data for this study.

6.2.4. Experimental setup

Network

We obtain the main road network between the incident location and the Netherlands from OpenStreetMap via the OSMnx Python library (Boeing, 2017). Only motorways, and primary and secondary roads are imported, resulting in a road network with 78 034 nodes, of which 17 839 in the Netherlands. The same road network is used for all cases.

Simulation of escape routes

For each of the three cases, 1 000 escape routes are simulated to random points in three large cities in the Netherlands (Amsterdam, Rotterdam, and Utrecht). We use the shortest path model with 2% noise, which is also used in Chapters 3 and 5.

Optimization of interception positions

The police interception positions are optimized using the static sequential simulation-optimization model proposed in Chapter 2 and used throughout this dissertation. The model optimizes the police positions to maximize the number of intercepted routes.

The police interception positions are limited to the eastern provinces of the Netherlands: Overijssel, Gelderland, Noord-Brabant, and Limburg. Dutch police cannot make arrests outside the Netherlands. The western provinces are excluded to prevent interception positions at escape nodes, which would be an artifact of the assumptions underlying the simulation model of the fugitive and ineffective for real-world interception. This constraint results in 9,995 possible nodes for interception.

The dataset does not include the locations of the police units at the time of the incidents. In other words, we do not know the starting positions of the police units for the optimization and evaluation of interception positions. Therefore, we assume the police units' starting positions to be the locations they were positioned at by the control room. Given the long time between the incident report and the suspect crossing into the Netherlands, we do not expect this assumption to have a large effect on the results of this study.

The model is optimized using the genetic algorithm used throughout this dissertation, for 100 000 function evaluations, and 5 random seeds to account for the seed variability in the optimization algorithm.

METRICS FOR EVALUATION

To evaluate the calculated police interception positions, we look at the effectiveness and efficiency of the positions.

- The *effectiveness* is best-found solution across seeds. The resulting percentage of intercepted routes is compared to the percentage of routes intercepted by the real police positions.
- To evaluate the *efficiency*, we examine the percentage of routes intercepted by fewer police units. Initially, we tried evaluating efficiency through a bi-objective problem formulation, optimizing both the number of police units used and the number of intercepted routes. However, this problem formulation did not converge. Instead, we evaluate the efficiency using the single-objective problem formulation, which outperforms the bi-objective solutions For each seed, we:

  1. Recursively remove police units where their removal does not impact the solution quality. This removes the redundancy in police positions when multiple police units are positioned on the same route.
  2. For each combination of the remaining interception positions, calculate the percentage of intercepted routes. Record the number of positions as the number of utilized police units.

  For each number of utilized police units, retain only the combination of positions that result in the highest solution quality across combinations and seeds.

6.3. ANALYSIS OF CASES

6.3.1. FIRST CASE

The first case is an ATM raid in the southwest of Germany (Rhineland-Palatinate), where the suspects fled towards the Netherlands. Nine police units were positioned to intercept the suspects.

SIMULATION OF FUGITIVE ROUTES

Figure 6.1 shows the simulated routes from the location of the ATM raid in southwest Germany to three major cities in the Netherlands. The routes primarily concentrate on 5 major highways in Germany and disperse in the Netherlands. The simulated routes include the location where the fleeing suspects were first spotted, indicating that the routes could be realistic.

EFFECTIVENESS OF POLICE INTERCEPTION POSITIONS

Figure 6.2 shows a comparison of the real and calculated police interception positions. The real positions are clustered around a few areas, and local police units are posted in their area of operation. The simulated routes are more spread out than the real police positions seem to anticipate. Due to the concentrated positions of the police units, only 57.3% of the simulated routes are intercepted.

In contrast, the calculated positions are more dispersed, with police units positioned at funnel locations where multiple simulated routes converge. The calculated positions intercept 87.7% of the simulated escape routes.

EFFICIENCY OF POLICE INTERCEPTION POSITIONS

Figure 6.3 shows the trade-off between the number of police units positioned to intercept the fugitive and the percentage of simulated routes intercepted. The figure shows that the same percentage of intercepted routes can be achieved with 8 police units instead of 9. Even with just 5 units, 85.9% of routes can be intercepted, which is 98% effectiveness compared to the best-found solution. With fewer than 5 utilized police units, the percentage of intercepted routes quickly tapers off to 39.6% with 1 unit, and, of course, 0% with no police units.

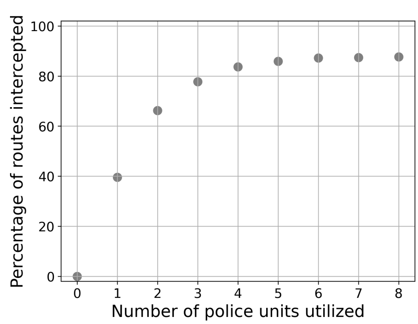


Figure 6.3: Trade-off between the number of police units used for interception and the percentage of routes intercepted for case 1.

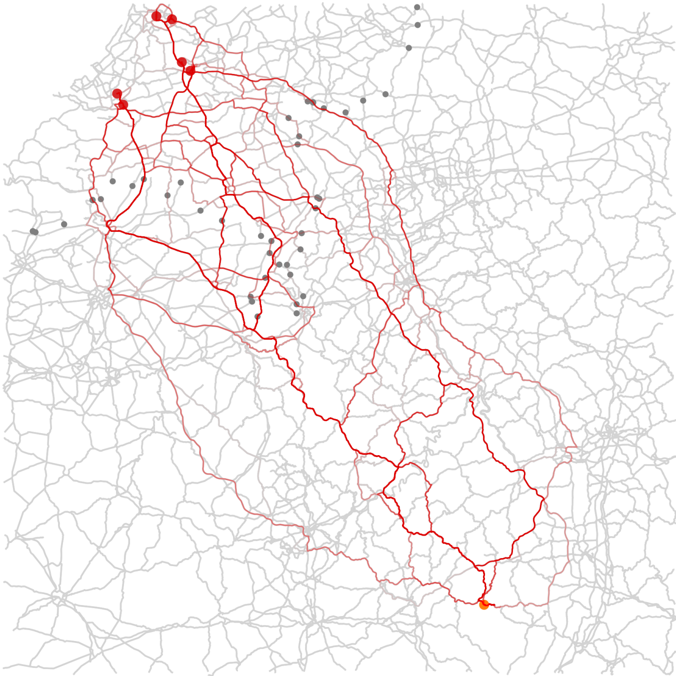
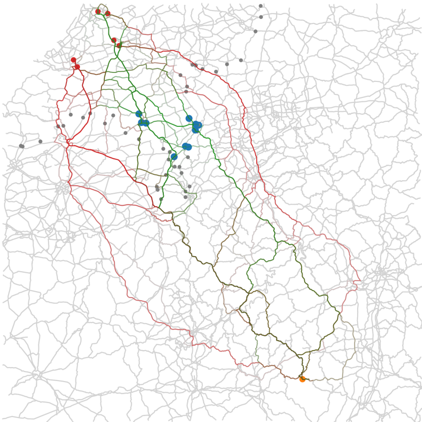
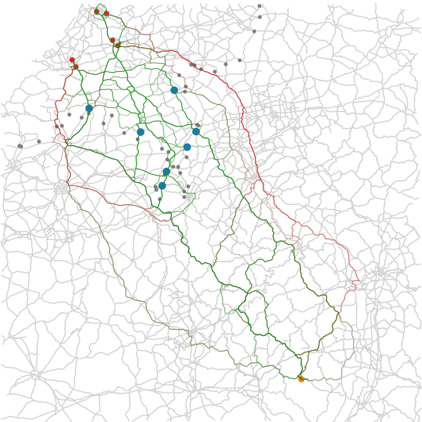


Figure 6.1: The simulated routes for case 1 from the location of the incident (orange) to random end nodes in three major cities in the Netherlands (red). The thickness of the road segment indicates the density of routes. As a visual aid, the grey nodes are the border crossings to the Netherlands.



(a) Real interception positions. Percentage of routes intercepted: 57.3%.



(b) Calculated interception positions. Percentage of routes intercepted: 87.7%.

Figure 6.2: The interceptions of the real and calculated police positions for case 1. Green indicates intercepted and red indicates not intercepted routes. The blue dots are the police positions.

6.3.2. SECOND CASE

The second case is an ATM raid in the west of Germany (North Rhine-Westphalia), where the suspects fled towards the Netherlands. Sixteen police units were positioned to intercept the suspects.

SIMULATION OF FUGITIVE ROUTES

Figure 6.4 shows the simulated routes from the location of the ATM raid in West Germany to three major cities in the Netherlands. Most routes cross the border to the Netherlands around Nijmegen, with a few diverging further North or South. In this case, the fleeing suspects were not spotted, so we cannot check if this location is included in the simulated routes.

EFFECTIVENESS OF POLICE INTERCEPTION POSITIONS

Figure 6.5 shows a comparison of the real and calculated police interception positions. The real positions are concentrated on the most likely road from the incident start location to the West of the Netherlands, intercepting a large percentage of routes. However, for this evaluation, the redundancy of 5 police units on a single road is not reflected in the measure of the effectiveness of the interception. Additionally, the real police positions are focused more on the North than the simulated routes. Importantly, the simulated routes reveal gaps in the interception positions, leaving important roads to the Netherlands exposed. The calculated positions cover all border crossings to the Netherlands, intercepting 100% of simulated routes.

6

EFFICIENCY OF POLICE INTERCEPTION POSITIONS

Figure 6.6 shows the trade-off between the number of police units positioned to intercept the fugitive and the percentage of simulated routes intercepted. The figure shows that the same percentage of intercepted routes can be achieved with 11 police units instead of 16, a significant decrease. With a decreasing number of police units, the percentage of intercepted routes slowly tapers off. Most routes (73.9%) cross the border near Nijmegen, which is also where the real positions are concentrated. In the optimization, these routes are intercepted by just 1 police unit.

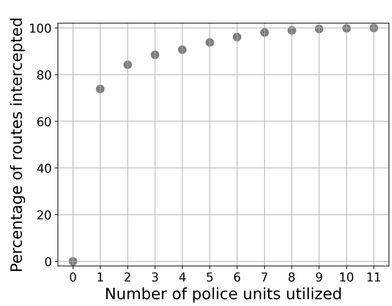


Figure 6.6: Trade-off between the number of police units used for interception and the percentage of routes intercepted for case 2.

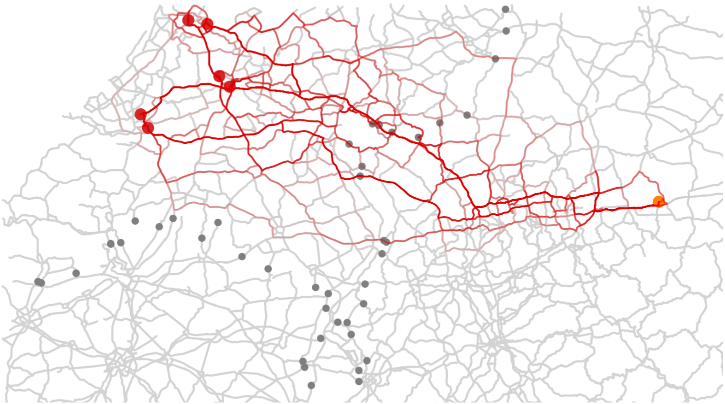
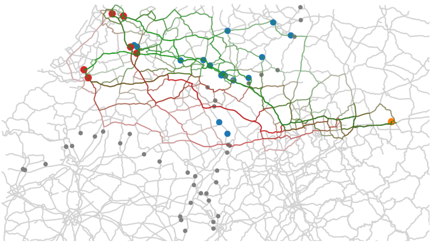
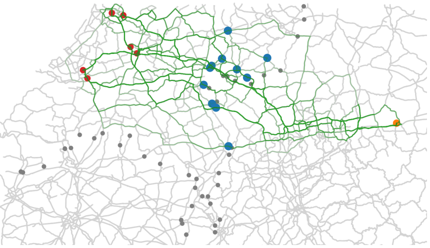


Figure 6.4: The simulated routes for case 2 from the location of the incident (orange) to random end nodes in three major cities in the Netherlands (red). The thickness of the road segment indicates the density of routes. As a visual aid, the grey nodes are the border crossings to the Netherlands.



(a) Real interception positions. Percentage of routes intercepted: 84.6%.



(b) Calculated interception positions. Percentage of routes intercepted: 100.0%.

Figure 6.5: The interceptions of the real and calculated police positions for case 2. Green indicates intercepted and red indicates not intercepted routes. The blue dots are the police positions.

6.3.3. THIRD CASE

The third case is an ATM raid in the west of Germany and close to the Netherlands (North Rhine-Westphalia), where the suspects fled towards the Netherlands. The police response differs significantly from that in the second case, making it an interesting case to examine. Twenty-two police units were positioned to intercept the suspects.

SIMULATION OF FUGITIVE ROUTES

Figure 6.7 shows the simulated routes from the location of the ATM raid in West Germany to three major cities in the Netherlands. Most routes cross the border to the Netherlands around Nijmegen. Because the incident occurred closer to the border, the routes diverge less compared to the second case. The simulated routes include the location where the fleeing suspects were first spotted, indicating that the routes could be realistic.

EFFECTIVENESS OF POLICE INTERCEPTION POSITIONS

The real interception positions for case 3 cover the border between the Netherlands and Germany. The simulated routes reveal two border crossings that are not covered, leading to 94.5% of simulated routes being intercepted. The calculated positions are much more concentrated but intercept 100% of simulated routes.

EFFICIENCY OF POLICE INTERCEPTION POSITIONS

Figure 6.9 shows the trade-off between the number of police units positioned to intercept the fugitive and the percentage of simulated routes intercepted. The figure shows that the same percentage of intercepted routes can be achieved with 7 police units instead of 22, a significant decrease. Again, the majority of the routes (85.4%) converge around Nijmegen and can be intercepted by a single police unit.

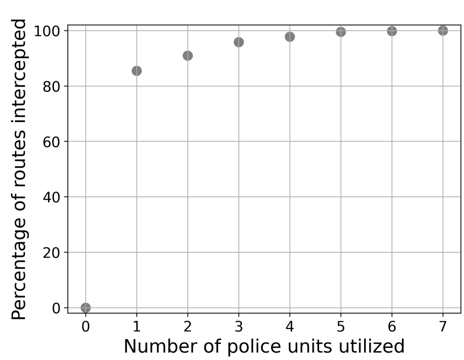


Figure 6.9: Trade-off between the number of police units used for interception and the percentage of routes intercepted for case 3.

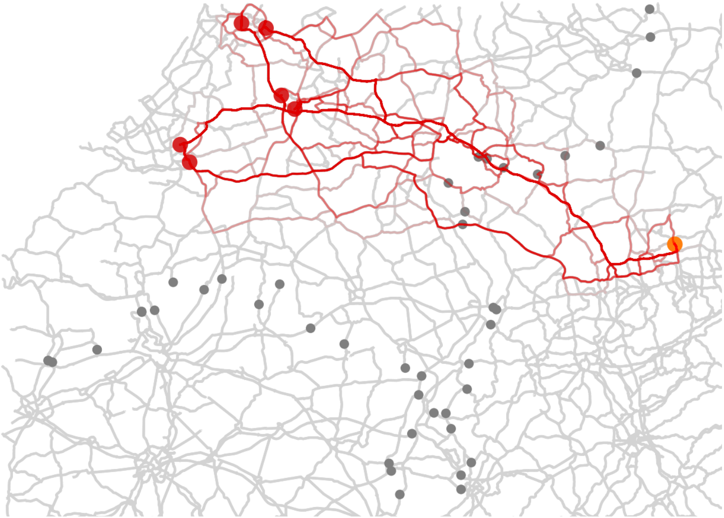
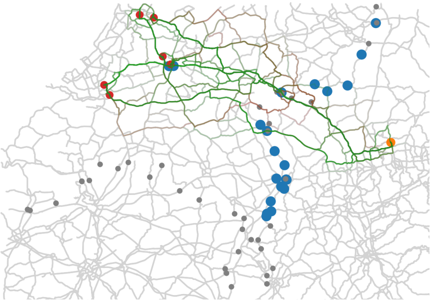
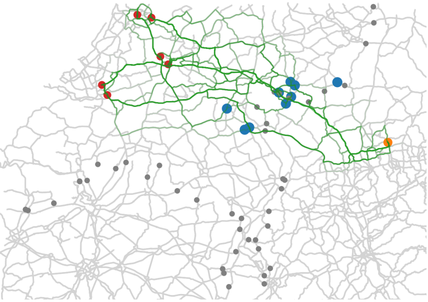


Figure 6.7: The simulated routes for case 3 from the location of the incident (orange) to random end nodes in three major cities in the Netherlands (red). The thickness of the road segment indicates the density of routes. As a visual aid, the grey nodes are the border crossings to the Netherlands.



(a) Real interception positions. Percentage of routes intercepted: 94.4%.



(b) Calculated interception positions. Percentage of routes intercepted: 100.0%.

Figure 6.8: The interceptions of the real and calculated police positions for case 3. Green indicates intercepted and red indicates not intercepted routes. The blue dots are the police positions.

6.4. DISCUSSION

In this chapter, we compared the calculated and real police interception positions using three case studies. This section discusses potential benefits of using models for decision support in the control room, explains the model assumptions that contribute to differences between the real and calculated interception strategies, and suggests areas for model improvements and further research.

The experiments showed that in two cases, the locations where fleeing fugitives were spotted were included in the simulated escape routes, suggesting that the routes could be realistic. While this alone does not validate the routes, the inclusion of these locations provides initial evidence of their plausibility. Furthermore, the third case demonstrates that even with an effective interception strategy, simulating escape routes and evaluating the strategy based on these routes can reveal gaps. This could be highly valuable in the control room.

6.4.1. USE OF POLICE INFORMATION

The real police interception positions are informed by additional information about the suspects and former experiences with similar types of crime. On the one hand, this could result in a more effective interception strategy. On the other hand, this leads to a focus on known patterns, while criminals often change their tactics, especially when the police start catching on (Bowers & Johnson, 2003; Reppetto, 1976). Research in other planning contexts shows that relying on the status quo leads to strategies that are not robust, making them ineffective in scenarios that differ from historical trends and expected future scenarios (Bankes, 1993). The simulation-optimization framework proposed in this dissertation is flexible and can incorporate different models of fugitive behavior. It would also be valuable to explore ways of enabling interactive route adjustments to incorporate centralist input.

6.4.2. REDUNDANT INTERCEPTION POSITIONS

Additionally, the real police positions are more redundant than the calculated positions. This is caused by two model assumptions. First, the optimization model assumes a 100% probability of detection, meaning a route is intercepted if a police unit is positioned on the route. However, in reality, the fleeing suspect could slip through and avoid detection, especially on busy roads where they can blend in with traffic. Second, the model considers a route intercepted by just standing there. However, in practice, an initial sighting typically leads to a pursuit, requiring multiple police vehicles to carry out an arrest by boxing in the suspect (Algemeen Dagblad, 4 February 2020). Positioning multiple units close together could increase the probability of arrest, though the optimization algorithm currently treats these additional units as redundant.

6.4.3. OPERATING AREA

Police units are generally limited to specific operating areas, which is reflected in real interception positions, where local units are stationed near cities like Utrecht and Eindhoven. However, the calculated positions in the model do not account for this constraint, which can result in local units being assigned outside their operating area. Addressing this is straightforward: during the search space representation step of the optimization, the possible interception positions for each unit can be filtered to stay within their operating areas (see Chapter 3).

6.5. CONCLUSION

This chapter evaluates the models proposed in this dissertation by comparing the real and calculated police interception positions for three case studies. The analysis showed that:

- The simulated fugitive routes include the locations at which the fugitive was spotted, suggesting that the routes are not unrealistic.
- The simulation of the fugitive escape routes reveals gaps in the police interception positions and could therefore assist the control room to improve the police interception strategy.
- The optimization of police interception positions could improve the efficiency of the interception strategy by reducing the number of police units involved.
- The simulated routes (and therefore the calculated interception positions) differ from the real police strategy; in some cases, they are more dispersed than the police interception positions, and in others, they are more concentrated. However, we do not know whether this discrepancy is due to the limited behavioral detail in the escape route simulator or to a narrow police strategy.
- The differences between the real and calculated police interception positions can be partly attributed to the underlying model assumptions, particularly the probability of detection, the operating area of the intercepting police units, and the need for multiple police units for arrest.

Additional experiments should investigate whether these conclusions apply to other types of incidents and fugitive behavior. Future research should also explore the reasons behind the differences between real and calculated positions by interviewing control room centralists. These interviews could be conducted retrospectively for historical cases or by using fictional cases to compare centralist and calculated strategies.

7

DISCUSSION

This chapter reflects on the methodology and practical aspects of the dissertation, discusses the generalizability of the results, and provides recommendations for future research.

7.1. METHODOLOGICAL REFLECTION

This section explores how real-world complexities that were not considered in this research might impact the methods and results of this dissertation. The complexities are categorized into fugitive behavior, police behavior, interactions between fugitives and police, information updates, the road network, traffic, and optimization.

7.1.1. FUGITIVE BEHAVIOR

First, models of fugitive behavior could simulate escape routes through more dense road networks. Chapters 2 and 3 show that when the number of nodes in the network increases, and when fugitive routes become more dispersed, the computation time for the optimization problem also increases. Therefore, while the simulation model in the simulation-optimization framework could be adapted to use any fugitive behavior model, other models of fugitive behavior could affect the timeliness of the optimization.

Second, fleeing fugitives may use other modes of transport, such as scooters, mopeds, public transit, or even travel on foot. Considering these modes requires a road network that includes bike and pedestrian paths, and public transit lines and schedules. While it is possible and quite easy to include this in the fugitive simulation model, extending the road network increases computation time, threatening the timeliness of the optimization. Additionally, it becomes important to differentiate between types of police vehicles — car, motorcycle, and bicycle — in terms of speed and road segments they can access.

Third, fugitives may plan to swap vehicles during their escape, which significantly increases the number of possible routes. To manage this, it is important to research locations that are likely for vehicle swaps, considering factors like camera avoidance and parking availability. Besides affecting simulated routes, a vehicle swap affects information updates. Chapter 5 assumes that the vehicle's license plate is known, but after a swap, the new plate may not be, leading to information updates for only a part of escape routes. The tested methods could easily be adapted to handle vehicle swaps by adding a condition to the detection logic.

Fourth, this dissertation considers cases with a single fugitive car, but in many situations, suspects split into multiple cars. The static optimization in this dissertation can easily handle multiple fugitives by simulating multiple sets of routes, which could even be based on different behavioral models. Periodic Re-Optimization, Direct Policy Search and Policy Tree Optimization could all be applied to dynamic cases that incorporate information updates.

7.1.2. POLICE BEHAVIOR

First, this dissertation assumes a 100% probability of detection, meaning that a route is fully intercepted if a police unit is positioned on it. However, in reality, the fleeing suspect could slip through and avoid detection, especially on busy roads where they can blend in with traffic. Chapter 6 showed that this assumption leads to very different interception strategies. The methods used in this dissertation could be adapted to consider probabilistic detection. Instead of simply counting routes, each route would be assigned a total score of 1 and each police unit intercepting the route would capture 80% of that score. A second unit intercepting the same route would capture 80% of the remaining 20% (i.e., 16%, resulting in a cumulative 96%). This percentage could also depend on factors such as road congestion. However, this approach would require more time per function evaluation, as it involves a different way of accounting for intercepted routes when calculating the quality of a solution.

Second, police units are generally limited to specific operating areas. The methods used in this dissertation do not account for this constraint, which can result in local units being assigned outside their operating area (see also Chapter 6). Addressing this is straightforward: during the search space representation step of the optimization, the possible interception positions for each unit can be filtered to stay within their operating areas (see Chapter 3).

7.1.3. INTERACTION EFFECTS

This dissertation assumes that the fugitive and police do not influence each other. However, in reality, the fugitive may react to police, for example when hearing sirens. Modeling these interactions would require simulation model optimization, where the simulation model describes both the fugitive and the police, and their interactions. Chapter 2 showed that simulation model optimization is significantly slower than sequential simulation-optimization, where the fugitive and police are decoupled, and which is used throughout this dissertation.

7.1.4. TRAFFIC

The interaction with traffic affects the police and the fugitive in different ways. The police, using sirens, can maneuver through traffic more easily by accessing the shoulder, running red lights, and cars get out of the way. On the other hand, the fugitive is more likely to encounter delays and get stuck in traffic or risks detection by breaking traffic laws, such as running red lights or weaving through vehicles. Zuurdeeg (2024) show that these differences in how each party interacts with traffic can significantly influence the effectiveness of police interception. Incorporating static traffic into the travel time calculations for both fugitives and police is straightforward to implement and has no impact on the computation time. Further research should extend these experiments by adding dynamic real-time traffic information. However, including dynamic traffic does affect the timeliness of the optimization.

7.1.5. INFORMATION UPDATES

First, this dissertation assumes a 100% probability of detection by ANPR cameras, meaning that the fugitive is always detected when passing by a camera. However, in reality, the fleeing suspect could avoid detection by hiding behind trucks, and camera performance is impaired at high speeds or in bad weather conditions (van Berkel et al., 2021). As a result, both false positives (where the fugitive is detected by a camera but was not actually there) and false negatives (where the fugitive passes a camera but is not detected) occur. Among the adaptive methods tested in this dissertation, Periodic Re-optimization can handle imperfect detections quite easily, similar to how it would handle imperfect interception (Section 7.1.2). For Direct Policy Search and Policy Tree Optimization, the imperfect detections should be implemented in the evaluation step, where the candidate interception strategy is tested across realizations of the fugitive routes. Introducing this noise would likely slow convergence significantly, threatening the timeliness of the optimization.

Second, the police uses civilian reports of abnormal behavior (e.g., driving on the wrong side of the road or at excessive speeds) to inform their interception strategy. In such cases, the detection location is not fixed and can occur anywhere on the road network at any time. Pre-computed adaptive approaches cannot account for this, as the state space would explode in size, making real-time calculation of interception strategies infeasible. For periodic re-optimization, such detections would either change the relative rewards of different simulated escape routes or simply exclude routes for the next re-optimization.

Third, the police can use helicopters or drones to scout for the fugitive. The routing of helicopters or drones should be optimized alongside the positions of the police vehicles. While promising, this requires extensive further research to develop effective methods.

7.1.6. ROAD NETWORK

Throughout this dissertation, we have seen a dependence of results on the topology of the road network. First, it is apparent that it is much easier to intercept a fleeing fugitive on some road networks than others. Sparser road networks, or those where funnels of fugitive routes emerge lead to higher probability of interception compared to complex (city) road networks (Chapter 4). Second, the size and topology of the road network impact the computation time required for the optimization and the extent to which search space representation can reduce the size of the network (Chapter 3). Combining coarsening approaches - especially further preprocessing the road network to remove unimportant parts of the network - could further reduce computation time. Third, the topology of the road network determines how easily police units can change interception strategies, and therefore how important flexibility to adapt to information updates is for the effectiveness of the interception strategy (Chapter 5). Understanding the relationship between network topology (measures) and the effectiveness of interception strategy and algorithm performance would improve the applicability of the methods in this dissertation to support the police control room.

7.1.7. OPTIMIZATION

This dissertation uses a genetic algorithm to optimize the simulation-optimization models, a choice based on its proven effectiveness in the literature, open-source availability, and preliminary experimentation comparing different algorithms. Throughout this research, it has become clear that the fugitive interception problem is difficult to solve, especially given the time constraints. Future research should compare different (configurations of) optimization algorithms, and perhaps develop a tailored heuristic, to improve real-time performance.

7.2. PRACTICAL REFLECTION

This section describes the practical steps to be taken before applying the models for decision support in the police control room. The steps are categorized into validation, decision support, fugitive behavior, police behavior, and the road network.

7.2.1. VALIDATION

To apply models in the police control room to advise on interception strategies, the models need to be thoroughly validated and tested. We suggest at least three approaches:

1. *Historical data:* While Chapter 6 compares the real and calculated positions for three ATM raids, additional analyses should examine other types of incidents and fugitive behavior. This should include interviews with control room centralists to understand the reasons for differences between real and calculated interception positions.

2. *Prototypical cases:* Discussing prototypical cases to compare and discuss interception positions could be a less time-consuming analysis than using historical data. It also allows for exploring cases that are poorly reflected in the existing dataset. Comparing the interception positions proposed by centralists with those calculated by the model can reveal aspects that might be over-

looked by either. In this way, the models could also serve as a training tool to help improve interception strategies, even if they are not directly used in real-time decision-making.

3. *Randomized controlled trials:* Lastly, testing the calculated police interception positions in cases with lower impact (e.g., for outstanding fines) could provide insight into further development needs and demonstrate the decision support system's added value. Assessing whether interception probability actually improves, however, is challenging because each incident is unique. Randomized controlled trials, as used in the medical field, could help compare the effectiveness of interception strategies when supported by the decision support system versus when they are not (Chalmers et al., 1981).

7.2.2. DECISION SUPPORT

First, this dissertation considers cases in isolation, whereas control room centralists have to weigh whether a case can be followed up, and if so, determine the appropriate number of police units to allocate. To support control room centralists in those decisions, it would be valuable to present the trade-off between the number of police units utilized and the probability of interception. The trade-off between keeping more police units available and adding a couple of percentage points to the interception should be made by the centralists, and depends on the severity of the case and the demand from other cases at that time.

Second, it would be valuable to extend the scope of the optimization to account for the concurrent and future demands on police (such as intercepting fleeing suspects, responding to emergencies like CPR, and other incidents). Extensive research has been done on the dynamic relocation of fire companies and ambulances. For example, Kolesar and Walker, 1974 look at how to maintain fire coverage when fire companies are already engaged in active firefighting, which can increase the risk of future fires because units are unavailable. van Barneveld et al., 2016 and van Barneveld et al., 2018 show the importance of dynamic ambulance relocations to maintain short response times and adapt to real-time demands. For fire companies and ambulances, the main challenge is relocating units to maintain coverage, with the responding vehicle and its destination fixed. In contrast, optimizing police interception requires balancing coverage - or how easily coverage can be restored - and the probability of intercepting the fleeing suspect.

Third, incorporating the experience and expertise of control room centralists in the interception strategy could improve its effectiveness. Input from centralists could be used to either refine the simulated routes — by including or excluding specific areas — or by adjusting the calculated interception strategy. One approach for collecting input is to present alternative interception strategies, allowing centralists to select the most suitable option; these diverse strategies can be generated using quality-diversity algorithms (Pugh et al., 2016). Another approach is interactive (multi-objective) optimization, where users can explore and engage with the solution space to tailor outcomes (Miettinen et al., 2016). Neither approach has been applied to fugitive interception and both should be evaluated for their real-time performance.

Fourth, interpretability (understanding what the outputs mean and how they can be used) and explainability (understanding how and why the model works) of the interception strategy could increase trust and acceptance of the decision support system (Shibl et al., 2013). Previous research on decision support systems for the Dutch Police found that it is crucial that individual police agents understand the reasoning behind the decision support and their contribution to the interception (Drenth & Steden, 2017; Herrewijnen et al., 2024). Future research should develop methods for improving the interpretability of the simulation of escape routes, the explainability of the optimization of the interception positions, and the interpretability of the presentation of the recommended interception positions.

7.2.3. FUGITIVE BEHAVIOR

For application in the control room, the library of fugitive behavior models should be expanded by interviewing control room centralists and police officers to develop profiles of fugitive behavior. However, criminals often change their tactics, especially when the police start catching on (Bowers & Johnson, 2003; Reppetto, 1976). Therefore, the collection of models should be continuously evaluated and extended to reflect current tactics.

7.2.4. ROAD NETWORK

The road network data used throughout this dissertation was obtained from OpenStreetMap, an open-source, crowd-sourced geographic database. However, errors occur in network topology and attributes, such as road classifications or speed limits. Additionally, we encountered a highway on-ramp that was mistakenly not connected to the main highway in the data. These inaccuracies affect the simulation of fugitive escape routes, the routing calculations for police units, and the coarsening of road networks. For the real-world application of models in the control room, the models should use the high-quality map data of the police.

7.3. GENERALIZABILITY

The fugitive interception problem is characterized by a difficult (NP-hard) network-based optimization problem that needs to be solved in very little time, where modeling behavior is important for the effectiveness of the optimization, and information gradually becomes available. Each chapter (2 - 5) focuses on different aspects of this problem, and each solves a methodological challenge.

- Chapter 2 addresses timely simulation-optimization, showing that separating controllable and uncontrollable components into optimization and simulation, respectively, leads to a significant reduction in the computation time. This approach, sequential simulation-optimization can be applied to other simulation-optimization problems where an optimal intervention has to be determined independent of the uncertainty in the system, such as autonomous vehicle control.

- Chapter 3 addresses graph coarsening and search space representation, which can be applied to various graph-based optimization problems to improve the timeliness of the optimization, such as large-scale route planning.
- Chapter 5 compares adaptive optimization approaches that have been applied to a variety of planning problems, including water management and climate adaptation. This comparison is valuable for understanding the contexts in which each approach is most effective, regardless of the application area.

In addition to the generalizability of the methods developed in this dissertation, the models can be applied to other graph-based search problems where policies or control actions must be determined quickly based on limited information. Examples are tracking down and managing virus outbreaks (Auping et al., 2017), wildfire protection and mitigation (Garcia-Martinez et al., 2015), and search and rescue (Hashimoto et al., 2022; Koester, 2008).

Furthermore, the research approach is applicable to other domains involving large-scale optimization problems on large networks, where rapid response times — within seconds or minutes — are essential. Examples are mitigating cascading failures in power networks (Smolyak et al., 2020) and rerouting train passengers after disruptions (Dollevoet et al., 2012). In both cases, responses to thousands of potential failure scenarios can be precomputed, enabling timely and effective decision-making during critical events.

7.4. FUTURE RESEARCH

This section discusses the topics for future research identified in the reflections. Following the structure of the reflections, we distinguish between topics for methodological research and those focused on practical application.

7.4.1. METHODOLOGICAL RESEARCH

Five interesting topics for future research emerged from the methodological reflections in Section 7.1. Below, we list these topics and, where relevant, discuss their potential implications for other application domains.

First, the accuracy of the models could be further improved by incorporating dynamic real-time traffic information (Subsection 7.1.4). However, this addition would increase the computation time for both simulating fugitive routes and optimizing police interception positions. Future research should evaluate the extent to which dynamic traffic information affects fugitive interception using a similar framework to Chapter 4 and Zuurdeeg (2024). If the impact proves significant, methods should be developed to integrate this information without significantly increasing computation time.

Second, this dissertation assumes that the information updates are accurate, while both false positives (where the fugitive is detected by a camera but is not there) and false negatives (where the fugitive passes a camera but is not detected) occur (Subsection 7.1.5). While imperfect information has been extensively studied in model predictive control and periodic re-optimization, further development and testing are needed for Direct Policy Search and Policy Tree Optimization. Improving these methods to handle imperfect information would be beneficial across applications, as real-world data is often unreliable or incomplete.

Third, the police can use helicopters or drones to scout for the fugitive. The routing of helicopters or drones should be optimized alongside the positions of the police vehicles. While promising, this requires extensive further research to develop effective and timely methods that can co-optimize the routing of helicopters or drones for scouting and police vehicles for intercepting (Subsection 7.1.5).

Fourth, we have seen a dependence of results on the topology of the road network throughout this dissertation (Subsection 7.1.6). Understanding the relationship between network topology measures and the effectiveness of optimization methods would allow the police to tailor the methods to specific cases or operating areas. Further research into network topology and fugitive interception would contribute to the broader fields of graph theory and network analysis.

Fifth, this dissertation uses a genetic algorithm to optimize the simulation-optimization models, a choice based on its proven effectiveness in the literature, open-source availability, and preliminary experimentation comparing different algorithms. Comparing a wide range of optimization algorithms for fugitive interception and developing heuristic methods could further improve timeliness and effectiveness (Subsection 7.1.7). These algorithms have applications beyond fugitive interception, including search and rescue, and, potentially, other graph-based optimization problems

7.4.2. PRACTICAL RESEARCH

Four interesting topics for future research emerged from the practical reflections in Section 7.2. Below, we list these topics and, where relevant, discuss their potential implications for other application domains.

First, the simulation of fugitive escape routes and calculated police interception positions should be validated and tested based on historical data, prototypical cases, and randomized controlled trials (Subsection 7.2.1). Besides improving the simulation and optimization, this research would provide valuable insights into validating decision support systems in critical, time-sensitive environments.

Second, interpretability and explainability of the interception strategy could increase trust and acceptance of the decision support system (Shibl et al., 2013). Future research should develop methods for improving the interpretability of the simulation of escape routes, the explainability of the optimization of the interception positions, and the interpretability of the presentation of the recommended interception positions. Research on the interpretability and explainability of simulation and optimization models could improve trust and broaden their applicability, particularly in sensitive and critical decision-making contexts.

Third, incorporating the experience and expertise of control room centralists in the interception strategy could improve its effectiveness. However, approaches that enable user input, such as interactive optimization, have yet to be applied to fugitive interception or time-constrained decision-making more generally. Future research should assess the real-time performance of these approaches and their impact on both the effectiveness of interceptions and trust in decision support systems.

Fourth, for application in the control room, the library of fugitive behavior models should be expanded based on interviews and analysis of historical data to develop profiles of fugitive behavior. To deal with crime displacement, the profiles and models should be continuously evaluated and extended to reflect current tactics.

8

CONCLUSION

Search and interception of fugitives by the police on a road network is a challenging task due to the complexity of the network, the unknown whereabouts of the fugitive and uncertainty about the routes that the fugitive takes, and time pressure: police control room centralists have, at most, a few minutes to decide where to position intercepting police units. Information technology, supported by modeling and simulation to depict the complex and stochastic decision space, can improve decision-making by suggesting interception positions for police units.

Simulation–optimization models are well-suited for real-time decision-support to the control room for search and interception of fugitives by police on a road network, due to their ability to encode complex behavior while still optimizing the interception. However, timely calculation of the recommended interception positions is essential to support police interception operations in real time. Given the complexity of the problem, caused by a large number of nodes in a road network, the uncertainty in the behavior of the fugitive, and the degrees of freedom of the police units, solving the simulation–optimization in real-time is challenging. Incorporating real-time information about the fugitive's location can improve decision-making, but it also makes the optimization problem even harder to solve within the limited time available.

This dissertation aims to identify, develop, and evaluate methods to identify effective fugitive interceptions. Four sub-research questions address the challenges in reaching this goal. Each research question aims to improve the effectiveness of the interception while preserving the timeliness of the calculated solutions. First, we formalize the fugitive interception problem. Second, to improve the timeliness of the optimization, we propose a search space representation method that reduces problem complexity without compromising solution quality. Third, we operationalize and compare different models of fugitive escape behavior. Fourth, we compare adaptive optimization approaches and outline the conditions under which each approach is most suitable. This chapter answers the research questions introduced in Chapter 1, and provides a general conclusion of the dissertation.

8.1. ANSWERING THE RESEARCH QUESTIONS

This dissertation identified, developed, and evaluated methods to identify effective fugitive interceptions. Four sub-research questions address the challenges in reaching this goal. In this section, we address each sub-question one by one and the next section presents a general conclusion.

1. *How to formalize fugitive interception?*

   We use simulation-optimization to model fugitive behavior and optimize police positioning. Fugitive routes are simulated to represent possible escape routes. The optimization is formalized as a variation of the flow interception problem, where the number of simulated routes intercepted by positioning police units is maximized. This formalization only considers interceptions at the end positions, not while units are en route. This simplification improves the optimization's timeliness; the flow interception problem is NP-hard, and adding routing would significantly increase complexity.

   Moreover, solving a typical simulation-optimization configuration in real-time is infeasible due to the complexity of the problem, caused by a large number of edges in a road network, the uncertainty in the behavior of the fugitive, and the degrees of freedom of the police units. A different way of combining simulation and optimization is sequential simulation-optimization, where the simulation constructs (part of) the constraints of an optimization problem. To answer this research question, Chapter 2 provides an extension to the taxonomy of simulation-optimization configurations, presents and researches sequential simulation-optimization, and provides a quantitative analysis of the real-time performance of classical simulation-optimization compared to sequential simulation-optimization. Thus, we show the potential of sequential simulation-optimization to mitigate the expensive optimization of simulation models. Additionally, the analysis shows that metaheuristic solution approaches reach a high quality of solutions in a fraction of the computation time of exact optimization algorithms. Experiments on a grid network and city road network demonstrate that these findings hold for various graph topologies. Sequential simulation-optimization and the metaheuristic solution approach are used to model fugitive interception throughout this dissertation.

2. *How to leverage graph coarsening to improve the timeliness of simulation-optimization for fugitive interception?*

   Graph coarsening, a technique to reduce the size of a graph while preserving essential structural properties, offers a promising approach to reducing the computation time of the fugitive interception problem. The effectiveness of graph coarsening algorithms varies depending on the application, as the importance of the nodes and links is very case-specific.

   Chapter 3 compares four graph coarsening techniques for fugitive interception across five road networks. Pruning – the removal of dead ends and self-loops – seems to always be effective: it removes 2.7% to 29.1% of nodes (depending on the network), but these nodes are likely not relevant for fugitive interception. Other preprocessed graph coarsening algorithms can significantly reduce the number of nodes in the networks, but cause the solution quality to deteriorate significantly. Important interception positions and paths for the police units are often not preserved for these algorithms. In contrast, on-the-fly network reconstruction, where a new network is created from the escape routes and the shortest paths from the police starting positions to any node on these escape routes, improves the optimization. By removing poor-quality solutions, the optimization algorithm converges more quickly and results in higher-quality solutions.

Based on these results, we propose an approach incorporating on-the-fly graph reconstruction into the Search Space Representation in the optimization process. This allows for more flexibility, capable of handling different fugitive profiles and network structures. Search space representation improves the quality of the best solutions obtained by the optimization algorithm with up to 12%. Notably, the reliability of the optimization to find high-quality solutions is increased: the average obtained solution quality across seed increases by up to 24%. Meanwhile, the number of function evaluations required to obtain high-quality solutions is reduced to 5 000 -10 000 depending on the size and complexity of the road network, which is feasible for real-time decision-making.

3. *How to generate an ensemble of realistic fugitive escape routes?*

Many theoretical studies (including the first version of the fugitive interception model in Chapter 2) implement a random motion for the fleeing suspect. Explicitly encoding behavior through decision rules could lead to more effective interception strategies.

To answer this research question, we conceptualize and operationalize two modes of fleeing suspect route choices in Chapter 4. We compare the resulting sets of routes and the optimized police interception positions. Finally, we evaluate the effectiveness of the police interception positions for different route generation models.

We found that knowledge of the specific route choice model of the fleeing suspect is critical for finding effective interception positions in complex networks with non-uniformly distributed features and obstacles. This paper conceptualizes and operationalizes three models of fleeing behavior to examine the resulting routes, calculated police interception positions, and the robustness of the models. We show that a random walk model - often used to simulate fleeing suspects in interception problems - leads to distinctly different escape routes and, therefore, calculated interception positions compared to models based on psychological theory. Therefore, a random walk model is unsuitable for decision support in real-world police interception. Despite their similarities in implementation, the Cool and Hot models result in different simulated escape routes and, therefore, calculated police interception positions. The differences are larger when the road network is complex and has non-uniformly distributed obstacles. The calculated interception positions are robust to different models of a fleeing suspect when the road network is either (1) relatively simple with few roads leading to the escape nodes, or (2) when police units can quickly reach intersections close to the incident, or (3) the positions of the escape nodes create a funnel where escape routes converge.

4. *How to use incoming information to increase the probability of interception?*

   Simulation–optimization can be used to support real-time decision-making for fugitive interception. Incorporating real-time information about the fugitive's location can improve decision-making, but it also makes the optimization problem harder to solve within the limited time available for real-time decision making.

   Chapter 5 examines the solution quality obtained for the fugitive interception problem with information updates within a limited number of function evaluations for four solution approaches: One-shot Optimization, Periodic Re-Optimization, Direct Policy Search, and Policy Tree Optimization. Our analysis shows that for the fugitive interception problem:

   (a) Adaptive solution approaches, utilizing the information updates, outperform static one-shot optimization, especially when the number of intercepting police units is low.
   (b) Direct Policy Search effectively avoids lock-ins and finds high-quality solutions across networks and problem instances. However, making the resulting solutions interpretable for decision-makers requires further research.
   (c) Policy Tree Optimization, while the most interpretable, converges too slowly for real-time decision support.
   (d) Periodic Re-Optimization performs well for networks and problem instances with few lock-ins (for instance, city networks) and is flexible to further extensions to include probabilistic interception and detection. While the optimization results at each time step are interpretable, Periodic Re-optimization does not offer insight into causal relationships between sensor information updates and changes in calculated positions of police units.

   Based on the results in this paper, practitioners are advised to use Direct Policy Search for problems that are vulnerable to lock-ins. If the interpretability of the results is critical, Direct Policy Search should be supplemented with an interpretable interface. Otherwise, practitioners are advised to use Periodic Re-optimization for its flexibility and ease of implementation.

8

8.2. General conclusion

This dissertation explores models for decision support to the police control room for fugitive interception. We highlight three main contributions. First, we demonstrate how sequential simulation-optimization reduces computation time compared to classical simulation model optimization. Additionally, we present a metaheuristic solution approach that identifies near-optimal solutions in a fraction of the time required for exact optimization, with the computation time increasing at a slower rate as the network size grows. Second, we identify a method for incorporating information updates — both observations and the absence of observations of the fugitive — into the interception strategy while maintaining consistency in the interception positions. Third, we show how behavioral assumptions impact the effectiveness of interception strategies. More detailed models of behavior can easily be incorporated into the proposed simulation-optimization method.

To summarize, this research provides the foundation for effective decision support to police control rooms to increase the chance of red-handed arrests.

Bibliography

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Chapter 10 - Metaheuristic algorithms: a comprehensive review. In A. K. Sangaiah, M. Sheng, & Z. Zhang (Eds.), *Intelligent data-centric systems* (pp. 185–231). Academic Press. https://doi.org/10.1016/B978-0-12-813314-9.00010-4

Algemeen Dagblad. (4 February 2020). Auto's rammen en 'inboxen', zo mag de politie een verdachte stoppen [https://www.ad.nl/tilburg/autos-rammen-en-inboxen-zo-mag-de-politie-een-verdachte-stoppen~ae9c0640/ Accessed: 2024-11-07].

Alós-Ferrer, C., Hügelschäfer, S., & Li, J. (2016). Inertia and ing. *Frontiers in Psychology*, *7*. https://doi.org/10.3389/fpsyg.2016.00169

Alspach, B. (2004). Searching and sweeping graphs: A brief survey. *Le Matematiche*, *59*, 5–37.

Amaran, S., Sahinidis, N. V., Sharda, B., & Bury, S. J. (2016). Simulation optimization: a review of algorithms and applications. *Annals of Operations Research*, *240*(1), 351–380. https://doi.org/10.1007/s10479-015-2019-x

Andradóttir, S. (1998). A review of simulation optimization techniques. In D. J. Medeiros, E. F. Watson, J. S. Carson, & M. S. Manivannan (Eds.), *Proceedings of the 1998 Winter Simulation Conference* (pp. 151–158). IEEE. https://doi.org/10.1109/wsc.1998.744910

Asano, T., & Hirata, T. (1983). Edge-contraction problems. *Journal of Computer and System Sciences*, *26*(2), 197–208. https://doi.org/https://doi.org/10.1016/0022-0000(83)90012-0

Auping, W. L., Pruyt, E., & Kwakkel, J. H. (2017). Simulating endogenous dynamics of intervention-capacity deployment: Ebola outbreak in Liberia. *International Journal of Systems Science: Operations & Logistics*, *4*(1), 53–67. https://doi.org/10.1080/23302674.2015.1128576

Azadivar, F. (1999). Simulation optimization methodologies. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, & G. W. Evans (Eds.), *Proceedings of the 1999 Winter Simulation Conference* (pp. 93–100). IEEE. https://doi.org/10.1145/324138.324168

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., & Mordatch, I. (2020). Emergent tool use from multi-agent autocurricula. *International Conference on Learning Representations*. https://arxiv.org/abs/1909.07528

Bakker, H., Dunke, F., & Nickel, S. (2020). A structuring review on multi-stage optimization under uncertainty: Aligning concepts from theory and practice. *Omega*, *96*, 102080. https://doi.org/10.1016/j.omega.2019.06.006

Bankes, S. (1993). Exploratory modeling for policy analysis. *Operations Research*, *41*(3), 435–449. https://doi.org/10.1287/opre.41.3.435

Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115. https://doi.org/https://doi.org/10.1016/j.inffus.2019.12.012

Bellman, R. (1966). Dynamic programming. *Science*, *153*(3731), 34–37. https://doi.org/10.1126/science.153.3731.34

Berman, O., Larson, R. C., & Fouska, N. (1992). Optimal location of discretionary service facilities. *Transportation Science*, *26*(3), 201–211. https://doi.org/10.1287/trsc.26.3.201

Boccia, M., Sforza, A., & Sterle, C. (2009). Flow intercepting facility location: Problems, models and heuristics. *Journal of Mathematical Modelling and Algorithms*, *8*(1), 35–79. https://doi.org/10.1007/s10852-008-9098-5

Bode, F., Reed, P. M., Reuschen, S., & Nowak, W. (2019). Search space representation and reduction methods to enhance multiobjective water supply monitoring design. *Water Resources Research*, *55*, 2257–2278. https://doi.org/10.1029/2018WR023133

Boeing, G. (2017). OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers Environment and Urban Systems*, *65*, 126–139. https://doi.org/10.1016/j.compenvurbsys.2017.05.004

Boeing, G. (2024). Modeling and analyzing urban networks and amenities with OSMnx. https://geoffboeing.com/publications/osmnx-paper/

Bopardikar, S. D., Bullo, F., & Hespanha, J. P. (2008). On discrete-time pursuit-evasion games with sensing limitations. *IEEE Transactions on Robotics*, *24*(6), 1429–1439. https://doi.org/10.1109/TRO.2008.2006721

Borie, R., Koenig, S., & Tovey, C. (2013). Section 9.5: Pursuit-evasion problems. In J. Gross, J. Yellen, & P. Zhang (Eds.), *Handbook of graph theory* (pp. 1145–1165). Chapman; Hall/CRC.

Borie, R., Tovey, C., & Koenig, S. (2011). Algorithms and complexity results for graph-based pursuit evasion. *Autonomous Robots*, *31*, 317–332. https://doi.org/10.1007/s10514-011-9255-y

Bowers, K. J., & Johnson, S. D. (2003). Measuring the geographical displacement and diffusion of benefit effects of crime prevention activity. *Journal of Quantitative Criminology*, *19*, 275–301. https://doi.org/10.1023/A:1024909009240

Carson, Y., & Maria, A. (1997). Simulation optimization: Methods and applications. In S. Andradóttir, K. J. Healy, D. H. Withers, & B. L. Nelson (Eds.), *Proceedings of the 1997 Winter Simulation Conference* (pp. 118–126). IEEE. https://doi.org/10.1145/268437.268460

Cesari, G., Schildbach, G., Carvalho, A., & Borrelli, F. (2017). Scenario model predictive control for lane change assistance and autonomous driving on highways. *IEEE Intelligent Transportation Systems Magazine*, *9*(3), 23–35. https://doi.org/10.1109/MITS.2017.2709782

Chalmers, T. C., Smith Jr., H., Blackburn, B., Silverman, B., Schroeder, B., Reitman, D., & Ambroz, A. (1981). A method for assessing the quality of a randomized control trial. *Controlled Clinical Trials*, *2*(1), 31–49. https://doi.org/10.1016/0197-2456(81)90056-8

Chevalier, C., & Safro, I. (2009). Comparison of coarsening schemes for multilevel graph partitioning. *Learning and Intelligent Optimization*, 191–205. https://doi.org/10.1007/978-3-642-11169-3_14

Chung, T. H., Hollinger, G. A., & Isler, V. (2011). Search and pursuit-evasion in mobile robotics: a survey. *Autonomous Robots*, *31*(4), 299–316. https://doi.org/10.1007/s10514-011-9241-4

Clarke, D., Mohtadi, C., & Tuffs, P. (1987). Generalized predictive control - part i. the basic algorithm. *Automatica*, *23*(2), 137–148. https://doi.org/10.1016/0005-1098(87)90087-2

Cohen, J. S., & Herman, J. D. (2021). Dynamic adaptation of water resources systems under uncertainty by learning policy structure and indicators. *Water Resources Research*, *57*(11). https://doi.org/10.1029/2021wr030433

Cormican, K. J., Morton, D. P., & Wood, R. K. (1998). Stochastic network interdiction. *Operations Research*, *46*(2), 184–197. https://doi.org/10.1287/opre.46.2.184

Cornish, D. B., & Clarke, R. V. G. (1986). *The reasoning criminal: Rational choice perspectives on offending*. Springer-Verlag, New York.

Cutler, C. R., & Ramaker, B. L. (1980). Dynamic matrix control - a computer control algorithm. *Proceedings of the 1980 Joint Automatic Control Conference*. https://doi.org/10.1109/JACC.1980.4232009

David P. Morton, F. P., & Saeger, K. J. (2007). Models for nuclear smuggling interdiction. *IIE Transactions*, *39*(1), 3–14. https://doi.org/10.1080/07408170500488956

de Armas, J., Juan, A. A., Marquès, J. M., & Pedroso, J. P. (2017). Solving the deterministic and stochastic uncapacitated facility location problem: From a heuristic to a simheuristic. *Journal of the Operational Research Society*, *68*(10), 1161–1176. https://doi.org/10.1057/s41274-016-0155-6

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. https://doi.org/10.1109/4235.996017

Delft High Performance Computing Centre. (2022). DelftBlue Supercomputer (Phase 1).

Dewinter, M., Vandeviver, C., Dau, P. M., Vander Beken, T., & Witlox, F. (2022). Hot spots and burning times : A spatiotemporal analysis of calls for service to establish police demand. *Applied Geography*, *143*, 13. https://doi.org/10.1016/j.apgeog.2022.102712

Dollevoet, T., Huisman, D., Schmidt, M., & Schöbel, A. (2012). Delay management with rerouting of passengers. *Transportation Science*, *46*(1), 74–89. https://doi.org/10.1287/trsc.1110.0375

Dong, N. A., Eckman, D. J., Zhao, X., Henderson, S. G., & Poloczek, M. (2017, December). Empirically comparing the finite-time performance of simulation-optimization algorithms. In W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, & E. Page (Eds.), *Proceedings of the 2017 Winter Simulation Conference* (pp. 2206–2217). IEEE. https://doi.org/10.1109/WSC.2017.8247952

Drenth, A., & Steden, R. (2017). Ervaringen van straatagenten met het criminaliteits anticipatie systeem. *Het Tijdschrift voor de Politie*, *79*, 6.

Dupačová, J., Consigli, G., & Wallace, S. W. (2000). Scenarios for multistage stochastic programs. *Annals of operations research*, *100*, 25–53. https://doi.org/10.1023/A:1019206915174

Eckman, D. J., Henderson, S. G., & Shashaani, S. (2023). Diagnostic tools for evaluating and comparing simulation-optimization algorithms. *INFORMS Journal on Computing*, *35*(2), 350–367. https://doi.org/10.1287/ijoc.2022.1261

Epstein, S., Pacini, R., Denes-Raj, V., & Heier, H. (1996). Individual differences in intuitive-experiential and analytical-rational thinking styles. *Journal of Personality and Social Psychology*, *71*(2), 390–405. https://doi.org/10.1037/0022-3514.71.2.390

Evans, J. (1984). Heuristic and analytic processes in reasoning. *British Journal of Psychology*, *75*, 451–468. https://doi.org/10.1111/J.2044-8295.1984.TB01915.X

Federal Bureau of Investigation. (2022). Crime in the United States: Offenses known to law enforcement.

Figueira, G., & Almada-Lobo, B. (2014). Hybrid simulation–optimization methods: A taxonomy and discussion. *Simulation Modelling Practice and Theory*, *46*, 118–134. https://doi.org/10.1016/j.simpat.2014.03.007

Fomin, F. V., & Thilikos, D. M. (2008). An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science*, *399*(3), 236–245. https://doi.org/10.1016/j.tcs.2008.02.040

Fu, M. C. (1994). Optimization via simulation: A review. *Annals of Operations Research*, *53*(1), 199–247. https://doi.org/10.1007/BF02136830

Fu, M. C. (2002). Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing*, *14*(3), 192–215. https://doi.org/10.1287/ijoc.14.3.192.113

Fu, M. C. (2015). Overview of the handbook. In M. C. Fu (Ed.), *Handbook of simulation optimization* (pp. 1–8). Springer.

Fu, M. C., Glover, F. W., & April, J. (2005). Simulation optimization: A review, new developments, and applications. In F. B. Armstrong, J. A. Joines, N. Steiger, & M. E. Kuhl (Eds.), *Proceedings of the 2005 Winter Simulation Conference* (pp. 83–95). IEEE. https://doi.org/10.1109/WSC.2005.1574242

Fukunaga, A. S. (1998). Restart scheduling for genetic algorithms. In A. E. Eiben, T. Bäck, M. Schoenauer, & H.-P. Schwefel (Eds.), *Parallel problem solving from nature* (pp. 357–366). Springer Berlin Heidelberg.

Garcia-Martinez, C., Blum, C., Rodriguez, F. J., & Lozano, M. (2015). The firefighter problem. *Computers & Operations Research*, *60*(100), 55–66. https://doi.org/10.1016/j.cor.2015.02.004

Garrison, W. T., & Petty, M. D. (2019). A comparison of simulation optimization algorithm performance. *Proceedings of SoutheastCon*, 1–7. https://doi.org/10.1109/SoutheastCon42311.2019.9020549

Geisberger, R., Sanders, P., Schultes, D., & Delling, D. (2008). Contraction hierarchies: Faster and simpler hierarchical routing in road networks. *International Workshop on Experimental and Efficient Algorithms*, 319–333. https://doi.org/10.1007/978-3-540-68552-4_24

Gendreau, M., Laporte, G., & Parent, I. (2000). Heuristics for the location of inspection stations on a network. *Naval Research Logistics*, *47*(4), 287–303. https://doi.org/10.1002/(SICI)1520-6750(200006)47:4<287::AID-NAV2>3.0.CO;2-R

Ghadimi, S., & Lan, G. (2015). Stochastic approximation methods and their finite-time convergence properties. In M. C. Fu (Ed.), *Handbook of simulation optimization* (pp. 179–206). Springer.

Giardini, F., Borit, M., Verhagen, H., & Wijermans, N. (2023). Modeling realistic human behavior in disasters. a rapid literature review of agent-based models reviews. In *Advances in Social Simulation. ESSA 2023*. Springer Nature Switzerland.

Giuliani, M., Castelletti, A., Pianosi, F., Mason, E., & Reed, P. M. (2016). Curses, tradeoffs, and scalable management: Advancing evolutionary multiobjective direct policy search to improve water reservoir operations. *Journal of Water Resources Planning and Management*, *142*(2), 04015050. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000570

Goharian, E., Shaltout, M., Erfani, M., & Eladawy, A. (2022). Developing an optimized policy tree-based reservoir operation model for high aswan dam reservoir, nile river. *Water*, *14*(7), 1061. https://doi.org/10.3390/w14071061

Gold, D. F., Reed, P. M., Gorelick, D. E., & Characklis, G. W. (2022). Power and pathways: Exploring robustness, cooperative stability, and power relationships in regional infrastructure investment and water supply management portfolio pathways. *Earth's Future*, *10*(2). https://doi.org/10.1029/2021EF002472

Gülpınar, N., Rustem, B., & Settergren, R. (2004). Simulation and optimization approaches to scenario tree generation. *Journal of Economic Dynamics and Control*, *28*, 1291–1315. https://doi.org/10.1016/S0165-1889(03)00113-1

Gupta, R., Hamilton, A., Reed, P., & Characklis, G. (2020). Can modern multi-objective evolutionary algorithms discover high-dimensional financial risk portfolio tradeoffs for snow-dominated water-energy systems? *Advances in Water Resources*, *145*, 103718. https://doi.org/10.1016/j.advwatres.2020.103718

Haasnoot, M., Kwakkel, J. H., Walker, W. E., & ter Maat, J. (2013). Dynamic adaptive policy pathways: A method for crafting robust decisions for a deeply uncertain world. *Global Environmental Change*, *23*(2), 485–498. https://doi.org/10.1016/j.gloenvcha.2012.12.006

Hadka, D., & Reed, P. M. (2013). Borg: An auto-adaptive many-objective evolutionary computing framework. *Evolutionary Computation*, *21*(2), 231–259. https://doi.org/10.1162/EVCO_a_00075

Hansen, P., Jaumard, B., & Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on Scientific and Statistical Computing*, *13*(5), 1194–1217. https://doi.org/10.1137/0913069

Hashimoto, A., Heintzman, L., Koester, R., & Abaid, N. (2022). An agent-based model reveals lost person behavior based on data from wilderness search and rescue. *Scientific Reports*, *12*, 5873. https://doi.org/10.1038/s41598-022-09502-4

Henderson, S. G. (2021). Reflections on simulation optimization. In S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, & M. Loper (Eds.), *Proceedings of the 2021 Winter Simulation Conference* (pp. 1–15). IEEE. https://doi.org/10.1109/WSC52266.2021.9715361

Henrichs, E., Lesch, V., Straesser, M., Kounev, S., & Krupitzer, C. (2022). A literature review on optimization techniques for adaptation planning in adaptive systems: State of the art and research directions. *Information and Software Technology*, *149*, 106940. https://doi.org/10.1016/j.infsof.2022.106940

Herman, J. D., & Giuliani, M. (2018). Policy tree optimization for threshold-based water resources management over multiple timescales. *Environmental Modelling & Software*, *99*, 39–51. https://doi.org/10.1016/J.ENVSOFT.2017.09.016

Herrewijnen, E., Loerakker, M. B., Vredenborg, M., & Woźniak, P. W. (2024). Requirements and attitudes towards explainable ai in law enforcement. *Proceedings of the 2024 ACM Designing Interactive Systems Conference*, 995–1009. https://doi.org/10.1145/3643834.3661629

Hodgson, M. J. (1990). A flow-capturing location-allocation model. *Geographical Analysis*, *22*, 270–279. https://doi.org/10.1111/j.1538-4632.1990.tb00210.x

Hong, L. J., Nelson, B. L., & Xu, J. (2015). Discrete optimization via simulation. In M. C. Fu (Ed.), *Handbook of simulation optimization* (pp. 9–44). Springer.

Ivić, S., Crnković, B., Arbabi, H., Loire, S., Clary, P., & Mezić, I. (2020). Search strategy in a complex and dynamic environment: The MH370 case. *Scientific Reports*, *10*, 19640. https://doi.org/10.1038/s41598-020-76274-0

Jacobs, P. H. M. (2005). *The DSOL simulation suite: Enabling multi-formalism simulation in a distributed context* [Doctoral dissertation, Delft University of Technology]. http://resolver.tudelft.nl/uuid:4c5586e2-85a8-4e02-9b50-7c6311ed1278

Kahneman, D. (2003). A perspective on judgment and choice: Mapping bounded rationality. *American Psychologist*, *58*(9), 697–720. https://doi.org/10.1037/0003-066x.58.9.697

Kehagias, A., Mitsche, D., & Prałat, P. (2014). The role of visibility in pursuit/evasion games. *Robotics*, *3*(4), 371–399. https://doi.org/10.3390/robotics3040371

Kempenaar, T. (2022). *Prospective criminal escape routes: An exploration of fugitive escape route decision-making using a dual-process approach* [Master's thesis, Delft University of Technology]. https://resolver.tudelft.nl/uuid:f3f734e3-3a5b-4fc7-8250-6099655b109c

Kennedy, W. G. (2012). Modelling human behaviour in agent-based models. In A. J. Heppenstall, A. T. Crooks, L. M. See, & M. Batty (Eds.), *Agent-based models of geographical systems* (pp. 167–179). Springer Netherlands. https://doi.org/10.1007/978-90-481-8927-4_9

Kim, S., Pasupathy, R., & Henderson, S. G. (2015). A guide to sample average approximation. In M. C. Fu (Ed.), *Handbook of simulation optimization* (pp. 207–243). Springer New York. https://doi.org/10.1007/978-1-4939-1384-8_8

Kleijnen, J., & Sargent, R. (2000). A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research*, *120*, 14–29. https://doi.org/10.1016/S0377-2217(98)00392-0

Koester, R. J. (2008). *Lost person behavior: A search and rescue guide on where to look for land, air, and water*. dbS Productions.

Kolesar, P., & Walker, W. E. (1974). An algorithm for the dynamic relocation of fire companies. *Operations Research*, *22*(2), 249–274. https://doi.org/10.1287/opre.22.2.249

Koutsoyiannis, D., & Economou, A. (2003). Evaluation of the parameterization-simulation-optimization approach for the control of reservoir systems. *Water Resources Research*, *39*(6). https://doi.org/10.1029/2003WR002148

Krishnakumari, P., Cats, O., & van Lint, H. (2020). Heuristic coarsening for generating multiscale transport networks. *IEEE Transactions on Intelligent Transportation Systems*, *21*(6), 2240–2253. https://doi.org/10.1109/TITS.2019.2912430

Lamouik, I., Yahyaouy, A., & Sabri, M. A. (2023). Model predictive control for full autonomous vehicle overtaking. *Transportation Research Record*, *2677*(5), 1193–1207. https://doi.org/10.1177/03611981221141432

Li, Y., He, R., & Guo, Y. (2007). An optimal model based on queueing theory for the problem of inspection station location in a traffic network. *Third International Conference on Natural Computation (ICNC 2007)*, *5*, 415–419. https://doi.org/10.1109/ICNC.2007.232

Loukas, A. (2019). Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, *20*(116), 1–42. http://jmlr.org/papers/v20/18-680.html

Maier, H. R., Razavi, S., Kapelan, Z., Matott, L. S., Kasprzyk, J., & Tolson, B. A. (2019). Introductory overview: Optimization using evolutionary algorithms and other metaheuristics. *Environmental Modelling & Software*, *114*, 195–213. https://doi.org/10.1016/j.envsoft.2018.11.018

Marangoni, G., Lamontagne, J., Quinn, J., Reed, P., & Keller, K. (2021). Adaptive mitigation strategies hedge against extreme climate futures. *Climatic Change*, *166*(3-4). https://doi.org/10.1007/s10584-021-03132-x

McPhail, C., Maier, H. R., Kwakkel, J. H., Giuliani, M., Castelletti, A., & Westra, S. (2018). Robustness metrics: How are they calculated, when should they be used and why do they give different results? *Earth's Future*, *6*(2), 169–191. https://doi.org/10.1002/2017EF000649

Mehlbaum, S., van Duijneveldt, I., Holvast, R., & van Arkel, D. (2014). Heterdaadkracht organiseren. *Tijdschrift voor de Politie*, *1*, 6–11.

Meneguzzer, C. (2023). Modeling the combined effect of travelers' contrarian behavior, learning and inertia on the day-to-day dynamics of route choice. *Applied Sciences*, *13*(5). https://doi.org/10.3390/app13053294

Miettinen, K., Hakanen, J., & Podkopaev, D. (2016). Interactive nonlinear multiobjective optimization methods. In S. Greco, M. Ehrgott, & J. R. Figueira (Eds.), *Multiple criteria decision analysis: State of the art surveys* (pp. 927–976). Springer New York. https://doi.org/10.1007/978-1-4939-3094-4_22

Moolenaar, D. E. G., Choenni, R., & van den Braak, S. W. (2023). *Criminaliteit en rechtshandhaving 2022* (tech. rep.). Wetenschappelijk Onderzoek- en Documentatiecentrum. https://www.rijksoverheid.nl/documenten/rapporten/2023/10/11/tk-bijlage-criminaliteit-en-rechtshandhaving-2022

Moussaïd, M., Helbing, D., & Theraulaz, G. (2011). How simple rules determine pedestrian behavior and crowd disasters. *Proceedings of the National Academy of Sciences*, *108*(17), 6884–6888. https://doi.org/10.1073/pnas.1016507108

Mühlenbein, H., Gorges-Schleuter, M., & Krämer, O. (1988). Evolution algorithms in combinatorial optimization. *Parallel Computing*, *7*(1), 65–85. https://doi.org/10.1016/0167-8191(88)90098-1

Murray, A., Matisziw, T., & Grubesic, T. (2007). Critical network infrastructure analysis: Interdiction and system flow. *Journal of Geographical Systems*, *9*, 103–117. https://doi.org/10.1007/s10109-006-0039-4

Nocedal, J., & Wright, S. (2006). *Numerical optimization* (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-40065-5

NOS Nieuws. (7 August 2017). Achtervolging na plofkraak in duitsland eindigt in huizen [https://nos.nl/artikel/2186883-achtervolging-na-plofkraak-in-duitsland-eindigt-in-huizen/ Accessed: 2024-11-04].

NOS Nieuws. (25 July 2017). Klopjacht in utrecht op verdachten na plofkraak in duitsland [https://nos.nl/artikel/2184900-klopjacht-in-utrecht-op-verdachten-na-plofkraak-in-duitsland/ Accessed: 2024-11-04].

Oliveira, R., & Loucks, D. P. (1997). Operating rules for multireservoir systems. *Water Resources Research*, *33*(4), 839–852. https://doi.org/10.1029/96WR03745

Parsons, T. D. (1978). Pursuit-evasion in a graph. In Y. Alavi & D. R. Lick (Eds.), *Theory and applications of graphs* (pp. 426–441). Springer. https://doi.org/10.1007/BFb0070400

Peleg, D., & Schäffer, A. A. (1989). Graph spanners. *Journal of Graph Theory*, *13*(1), 99–116. https://doi.org/10.1002/jgt.3190130114

Phillips-Wren, G., & Adya, M. (2020). Decision making under stress: The role of information overload, time pressure, complexity, and uncertainty. *Journal* *of Decision Systems*, *29*, 213–225. https://doi.org/10.1080/12460125.2020.1768680

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, *225*(1), 1–11. https://doi.org/10.1016/j.ejor.2012.08.015

Politie. (20 April 2024). Nederlands-duits succes in strijd tegen plofkraken [https://www.politie.nl/nieuws/2024/april/20/00-nederlands-duits-succes-in-strijd-tegen-plofkraken.html Accessed: 2024-11-06].

Powell, W. B. (2019). A unified framework for stochastic optimization. *European Journal of Operational Research*, *275*(3), 795–821. https://doi.org/https://doi.org/10.1016/j.ejor.2018.07.014

Psaraftis, H. (1980). A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, *14*, 130–154. https://doi.org/10.1287/trsc.14.2.130

Pugh, J. K., Soros, L. B., & Stanley, K. O. (2016). Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, *3*. https://doi.org/10.3389/frobt.2016.00040

Pung, J., D'Souza, R. M., Ghosal, D., & Zhang, M. (2022). A road network simplification algorithm that preserves topological properties. *Applied Network Science*, *7*(1). https://doi.org/10.1007/s41109-022-00521-8

Quinn, J. D., Reed, P. M., & Keller, K. (2017). Direct policy search for robust multi-objective management of deeply uncertain socio-ecological tipping points. *Environmental Modelling & Software*, *92*, 125–141. https://doi.org/10.1016/j.envsoft.2017.02.017

Raap, M., Preuß, M., & Meyer-Nieberg, S. (2019). Moving target search optimization - a literature review. *Computers and Operations Research*, *105*, 132–140. https://doi.org/10.1016/j.cor.2019.01.004

Ralphs, T. (2022). CBC: COIN-OR Branch-and-Cut solver [Version 2.10.8]. https://github.com/coin-or/Cbc

Reppetto, T. A. (1976). Crime prevention and the displacement phenomenon. *Crime & Delinquency*, *22*(2), 166–177. https://doi.org/10.1177/001112877602200204

Rodríguez-Flores, J. M., Gupta, R. S., Zeff, H. B., Reed, P. M., & Medellín-Azuara, J. (2023). Identifying robust adaptive irrigation operating policies to balance deeply uncertain economic food production and groundwater sustainability trade-offs. *Journal of Environmental Management*, *345*. https://doi.org/10.1016/j.jenvman.2023.118901

Rosenstein, M., & Barto, A. (2001). Robot weightlifting by direct policy search. *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*, *17*, 839–844.

Rothlauf, F. (2011). Design of modern heuristics: principles and application. In F. Rothlauf (Ed.), *Natural computing series* (pp. 45–102). Springer. https://doi.org/10.1007/978-3-540-72962-4_3

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, *1*(5), 206–215. https://doi.org/10.1038/s42256-019-0048-x

Rydzewski, A., & Czarnul, P. (2020). Recent advances in traffic optimisation: Systematic literature review of modern models, methods and algorithms. *IET Intelligent Transport Systems*, *14*(13), 1740–1758. https://doi.org/10.1049/iet-its.2020.0328

Safro, I., Sanders, P., & Schulz, C. (2015). Advanced coarsening schemes for graph partitioning. *Journal of Experimental Algorithmics*, *19*, 1–24. https://doi.org/10.1145/2670338

Sanders, P., & Schultes, D. (2012). Engineering highway hierarchies. *Journal of Experimental Algorithmics*, *17*. https://doi.org/10.1145/2133803.2330080

Sargent, R. G. (2011). Verification and validation of simulation models. In S. Jain, R. Creasey, J. Himmelspach, K. White, & M. Fu (Eds.), *Proceedings of the 2011 Winter Simulation Conference* (pp. 183–198). IEEE. https://doi.org/10.1109/WSC.2010.5679166

Sava, E., Twardy, C., Koester, R., & Sonwalkar, M. (2016). Evaluating lost person behavior models. *Transactions in GIS*, *20*, 38–53. https://doi.org/10.1111/tgis.12143

Schenato, L., Oh, S., Sastry, S., & Bose, P. (2005). Swarm coordination for pursuit evasion games using sensor networks. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. https://doi.org/10.1109/robot.2005.1570487

Shannon, R. E. (1998). Introduction to the art and science of simulation. In D. J. Medeiros, E. F. Watson, J. S. Carson, & M. S. Manivannan (Eds.), *Proceedings of the 1998 Winter Simulation Conference* (pp. 7–14). IEEE. https://doi.org/10.1109/WSC.1998.744892

Shapiro, A. (2003). Monte carlo sampling methods. In *Stochastic programming* (pp. 353–425, Vol. 10). Elsevier. https://doi.org/https://doi.org/10.1016/S0927-0507(03)10006-0

Sharma, A., Kosasih, E., Zhang, J., Brintrup, A., & Calinescu, A. (2022). Digital twins: State of the art theory and practice, challenges, and open research questions. *Journal of Industrial Information Integration*, *30*, 100383. https://doi.org/10.1016/j.jii.2022.100383

Shibl, R., Lawley, M., & Debuse, J. (2013). Factors influencing decision support system acceptance. *Decision Support Systems*, *54*(2), 953–961. https://doi.org/https://doi.org/10.1016/j.dss.2012.09.018

Shover, N., & Hochstetler, A. (2005). *Choosing white-collar crime*. Cambridge University Press. https://doi.org/10.1017/cbo9780511803482

Shukla, A., Pekny, J., & Venkatasubramanian, V. (2011). An optimization framework for cost effective design of refueling station infrastructure for alternative fuel vehicles. *Computers and Chemical Engineering*, *35*(8), 1431–1438. https://doi.org/10.1016/j.compchemeng.2011.03.018

Simard, F., Desharnais, J., & Laviolette, F. (2021). General cops and robbers games with randomness. *Theoretical Computer Science*, *887*, 30–50. https://doi.org/10.1016/j.tcs.2021.06.043

Simon, H. A. (1982). *Models of bounded rationality*. Cambridge: MIT Press.

Sinha, A., Malo, P., & Deb, K. (2018). A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, *22*(2), 276–295. https://doi.org/10.1109/tevc.2017.2712906

Skinner, G., & Parrey, B. (2019). A literature review on effects of time pressure on decision making in a cyber security context. *Journal of Physics: Conference Series*, *1195*, 012014. https://doi.org/10.1088/1742-6596/1195/1/012014

Smith, J. C., Prince, M., & Geunes, J. (2013). Modern network interdiction problems and algorithms. In P. M. Pardalos, D.-Z. Du, & R. L. Graham (Eds.), *Handbook of combinatorial optimization* (pp. 1949–1987). Springer New York. https://doi.org/10.1007/978-1-4419-7997-1_61

Smith, J. C., & Song, Y. (2020). A survey of network interdiction models and algorithms. *European Journal of Operational Research*, *283*(3), 797–811. https://doi.org/https://doi.org/10.1016/j.ejor.2019.06.024

Smolyak, A., Levy, O., Vodenska, I., Buldyrev, S., & Havlin, S. (2020). Mitigation of cascading failures in complex networks. *Scientific Reports*, *10*(1). https://doi.org/10.1038/s41598-020-72771-4

Stanovich, K. E., & West, R. F. (2000). Individual differences in reasoning: Implications for the rationality debate? *Behavioral and Brain Sciences*, *23*(5), 645–665. https://doi.org/10.1017/s0140525x00003435

Stone, L. D., Royset, J. O., & Washburn, A. R. (2016). Search for a stationary target. In *Optimal search for moving targets* (pp. 9–48). Springer International Publishing. https://doi.org/10.1007/978-3-319-26899-6_2

Tanaka, K., & Kurita, O. (2020). The probabilistic minisum flow interception problem: Minimizing the expected travel distance until intercept under probabilistic interception. *Geographical Analysis*, *52*(2), 211–230. https://doi.org/10.1111/gean.12197

Tolk, A. (2022). Simulation-based optimization: Implications of complex adaptive systems and deep uncertainty. *Information*, *13*(10), 469. https://doi.org/10.3390/info13100469

Tordecilla, R. D., Juan, A. A., Montoya-Torres, J. R., Quintero-Araujo, C. L., & Panadero, J. (2021). Simulation-optimization methods for designing and assessing resilient supply chain networks under uncertainty scenarios: A review. *Simulation Modelling Practice and Theory*, *106*, 102166. https://doi.org/10.1016/j.simpat.2020.102166

Torres, E. A., & Khuri, S. (2001). Applying evolutionary algorithms to combinatorial optimization problems. In V. N. Alexandrov, J. J. Dongarra, B. A. Juliano, R. S. Renner, & C. J. K. Tan (Eds.), *Proceedings of the International Conference on Computational Science* (pp. 689–700). Springer-Verlag.

Tutuarima, W. (2023). *Criminal fugitive escape routes: The influence of behavioural route-choice factors on criminal fugitive escape routes* [Master's thesis, Delft University of Technology]. https://resolver.tudelft.nl/uuid:236091cb-8d5c-4465-8f73-476dc7329d66

van Barneveld, T. C., Bhulai, S., & Van Der Mei, R. D. (2016). The effect of ambulance relocations on the performance of ambulance service providers. *European Journal of Operational Research*, *252*(1), 257–269. https://doi.org/10.1016/j.ejor.2015.12.022

van der Eng, T. (2018). Zo gaat de audi-bende te werk [https://reportersonline.nl/zo-gaat-de-audi-bende-te-werk/ Accessed: 2024-11-04].

van Barneveld, T., Jagtenberg, C., Bhulai, S., & van der Mei, R. (2018). Real-time ambulance relocation: Assessing real-time redeployment strategies for ambulance relocation. *Socio-Economic Planning Sciences*, *62*, 129–142. https://doi.org/10.1016/j.seps.2017.11.001

van Berkel, J., van Uden, A., & de Poot, C. (2021). *Evaluatie anpr-wetgeving 126jj wetboek van strafvordering* (tech. rep.). Wetenschappelijk Onderzoek- en Documentatiecentrum. https://www.rijksoverheid.nl/documenten/rapporten/2021/10/01/tk-bijlage-1-evaluatierapport-anpr-wetgeving-126jj

van Dijk, B., Terpstra, J. B., & Hulshof, P. (2013). *Heterdaadkracht in twee haagse pilotgebieden (in dutch)*. Politie & Wetenschap.

van Droffelaar, I. S., Kwakkel, J. H., Mense, J. P., & Verbraeck, A. (2022). Simulation-optimization configurations for fugitive interception. In B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C. G. Corlu, L. H. Lee, E. P. Chew, T. Roeder, & P. Lendermann (Eds.), *Proceedings of the 2022 Winter Simulation Conference*. IEEE. https://informs-sim.org/wsc22papers/pos133.pdf

van Droffelaar, I. S., Kwakkel, J. H., Mense, J. P., & Verbraeck, A. (2024a). The effect of models of criminal behavior on police interception strategies. *Advances in Social Simulation. ESSA 2024.*

van Droffelaar, I. S., Kwakkel, J. H., Mense, J. P., & Verbraeck, A. (2024b). Simulation-optimization configurations for real-time decision-making in fugitive interception. *Simulation Modelling Practice and Theory*, *133*, 102923. https://doi.org/10.1016/j.simpat.2024.102923

van Gelder, J.-L. (2013). Beyond rational choice: The hot/cool perspective of criminal decision making. *Psychology, Crime & Law*, *19*(9), 745–763. https://doi.org/10.1080/1068316x.2012.660153

Vanhée, L., & Borit, M. (2023). Thirty years of sense and sensibility in agent-based models: A bibliometric analysis. *Advances in Social Simulation. ESSA 2023.*

van Schilt, I. M., Kwakkel, J. H., Mense, J. P., & Verbraeck, A. (2022). Calibrating simulation models with sparse data: Counterfeit supply chains during covid-19. In B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C. G. Corlu, L. H. Lee, E. P. Chew, T. Roeder, & P. Lendermann (Eds.), *Proceedings of the 2022 Winter Simulation Conference* (pp. 496–507). https://doi.org/10.1109/WSC57314.2022.10015241

Walker, W., Haasnoot, M., & Kwakkel, J. (2013). Adapt or perish: A review of planning approaches for adaptation under deep uncertainty. *Sustainability*, *5*(3), 955–979. https://doi.org/10.3390/su5030955

Walker, W. E., Rahman, S., & Cave, J. (2001). Adaptive policies, policy analysis, and policy-making. *European Journal of Operational Research*, *128*(2), 282–289. https://doi.org/10.1016/S0377-2217(00)00071-0

Walshaw, C. (2004). A multilevel approach to the travelling salesman problem. *Operations Research*, *50*. https://doi.org/10.1287/opre.50.5.862.373

Washburn, A., & Wood, K. (1995). Two-person zero-sum games for network interdiction. *Operations Research*, *43*, 243–251. https://doi.org/10.1287/opre.43.2.243

Wetboek van Strafvordering, artikel 128. (1926). https://wetten.overheid.nl/jci1.3:c:BWBR0001903&boek=Eerste&titeldeel=VI&artikel=128

Wigan, M. R. (1972). The fitting, calibration, and validation of simulation models. *Simulation*, *18*, 188–192. https://doi.org/10.1177/003754977201800506

Xi, X., Sioshansi, R., & Marano, V. (2013). Simulation–optimization model for location of a public electric vehicle charging infrastructure. *Transportation Research Part D: Transport and Environment*, *22*, 60–69. https://doi.org/10.1016/j.trd.2013.02.014

Zanon, M., Frasch, J., Vukov, M., Sager, S., & Diehl, M. (2014). Model predictive control of autonomous vehicles. In H. Waschl, I. Kolmanovsky, M. Steinbuch, & L. del Re (Eds.), *Optimization and optimal control in automotive systems*. Springer.

Zatarain Salazar, J., Kwakkel, J. H., & Witvliet, M. (2023). Evaluating the choice of radial basis functions in multiobjective optimal control applications. *Environmental Modelling & Software*. https://doi.org/10.1016/j.envsoft.2023.105889

Zatarain Salazar, J., Reed, P. M., Herman, J. D., Giuliani, M., & Castelletti, A. (2016). A diagnostic assessment of evolutionary algorithms for multi-objective surface water reservoir control. *Advances in Water Resources*, *92*, 172–185. https://doi.org/10.1016/j.advwatres.2016.04.006

Zeigler, B., Prähofer, H., & Kim, T. G. (2000). *Theory of modeling and simulation* (2nd). Academic Press.

Zeng, W., Castillo, I., & Hodgson, M. J. (2010). A generalized model for locating facilities on a network with flow-based demand. *Networks and Spatial Economics*, *10*(4), 579–611. https://doi.org/10.1007/s11067-008-9073-8

Zuurdeeg, V. E. M. (2024). *Optimising fugitive interception: A comparative study into the added value of including more realistic traffic conditions in fugitive interception models* [Master's thesis, Delft University of Technology]. http://resolver.tudelft.nl/uuid:42796382-3ab9-4def-b6c8-773ac810f9f8

A

Appendix for Chapter 3: Tabular comparison of results

Tables A.1 - A.5 present the results of the evaluated graph coarsening algorithms for the five road networks. The results for node consolidation are presented using a tolerance value of 30 meters, as this value results in a balanced trade-off between node reduction and maintaining solution quality. For the heuristic coarsening algorithms, the maximum coarsening settings are applied, with pruning set to 1, iterations to the maximum, and the threshold also at its maximum value. For each algorithm, we report the minimum, median, mean, and maximum values across seeds. The first row in each table presents the results for the uncoarsened graph: Z is the fraction of intercepted routes and NFE is the number of function evaluations to search stall (reaching 95% of the solution quality). For each coarsening algorithm, the results are scaled to the best found for the uncoarsened graph: the maximum value for Z and the minimum value for NFE. A Z value close to or above 1 indicates good solution quality, while an NFE value below 1 indicates faster convergence.

Table A.1: Results of the coarsening algorithms for fugitive interception in Winterswijk.

		min	med	mean	max
G	Z	0.72	0.74	0.73	0.74
	NFE	3443	20433	27752	83163
Pruning	$Z(G_c)/Z(G)$	0.98	1.00	1.00	1.01
	$NFE(G_c)/NFE(G)$	0.62	5.43	6.27	18.00
Node consolidation	$Z(G_c)/Z(G)$	0.69	0.74	0.78	1.01
	$NFE(G_c)/NFE(G)$	0.41	2.88	3.04	6.22
Heuristic - type	$Z(G_c)/Z(G)$	0.34	0.34	1.00	0.35
	$NFE(G_c)/NFE(G)$	0.03	0.76	2.69	9.12
Heuristic - betweenness	$Z(G_c)/Z(G)$	0.34	0.34	0.34	0.36
	$NFE(G_c)/NFE(G)$	0.03	1.69	2.71	13.30
On-the-fly	$Z(G_c)/Z(G)$	0.99	1.00	1.00	1.01
	$NFE(G_c)/NFE(G)$	0.66	2.66	3.55	10.86

Table A.2: Results of the coarsening algorithms for fugitive interception in Manhattan.

		min	med	mean	max
G	Z	0.76	0.85	0.85	0.90
	NFE	23159	52635	52994	79822
Pruning	$Z(G_c)/Z(G)$	0.89	0.96	0.95	1.02
	$NFE(G_c)/NFE(G)$	1.19	1.72	2.00	3.12
Node consolidation	$Z(G_c)/Z(G)$	0.12	0.45	0.45	0.89
	$NFE(G_c)/NFE(G)$	0.31	1.09	1.73	4.15
Heuristic - type	$Z(G_c)/Z(G)$	0.01	0.42	0.48	1.00
	$NFE(G_c)/NFE(G)$	0.01	0.35	0.54	1.67
Heuristic - betweenness	$Z(G_c)/Z(G)$	0.01	0.01	0.05	0.30
	$NFE(G_c)/NFE(G)$	0.05	0.13	0.14	0.33
On-the-fly	$Z(G_c)/Z(G)$	0.91	0.97	0.96	0.99
	$NFE(G_c)/NFE(G)$	0.60	1.41	1.41	2.22

Table A.3: Results of the coarsening algorithms for fugitive interception in Utrecht.

		min	med	mean	max
G	Z	0.48	0.55	0.55	0.65
	NFE	24000	62978	58561	87309
Pruning	$Z(G_c)/Z(G)$	0.78	0.91	0.90	1.05
	$NFE(G_c)/NFE(G)$	0.86	2.36	2.32	3.25
Node consolidation	$Z(G_c)/Z(G)$	0.05	0.50	0.45	0.77
	$NFE(G_c)/NFE(G)$	0.66	1.78	2.13	4.14
Heuristic - type	$Z(G_c)/Z(G)$	0.00	0.00	0.01	0.03
	$NFE(G_c)/NFE(G)$	0.00	0.00	0.00	0.00
Heuristic - betweenness	$Z(G_c)/Z(G)$	0.80	1.00	0.99	1.06
	$NFE(G_c)/NFE(G)$	0.25	1.30	1.38	2.89
On-the-fly	$Z(G_c)/Z(G)$	0.75	0.90	0.90	1.05
	$NFE(G_c)/NFE(G)$	0.31	1.92	1.76	3.38

Table A.4: Results of the coarsening algorithms for fugitive interception in the main road network around Amsterdam.

		min	med	mean	max
G	Z	0.61	0.78	0.78	0.96
	NFE	5769	24928	36126	91322
Pruning	$Z(G_c)/Z(G)$	0.82	0.84	0.88	1.01
	$NFE(G_c)/NFE(G)$	1.53	5.00	6.68	17.34
Node consolidation	$Z(G_c)/Z(G)$	0.83	0.99	0.95	1.03
	$NFE(G_c)/NFE(G)$	0.30	8.01	7.23	15.50
Heuristic - type	$Z(G_c)/Z(G)$	0.00	0.00	0.00	0.01
	$NFE(G_c)/NFE(G)$	0.02	0.02	0.02	0.02
Heuristic - betweenness	$Z(G_c)/Z(G)$	0.00	0.02	0.34	0.89
	$NFE(G_c)/NFE(G)$	0.02	0.02	2.38	9.35
On-the-fly	$Z(G_c)/Z(G)$	0.84	1.01	0.99	1.03
	$NFE(G_c)/NFE(G)$	1.05	5.80	6.34	11.80

Table A.5: Results of the coarsening algorithms for fugitive interception in Rotterdam.

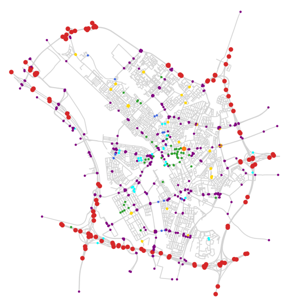
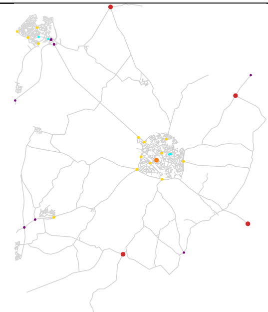
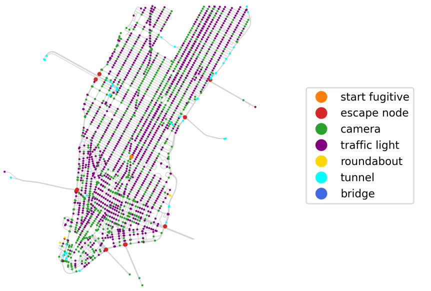
		min	med	mean	max
G	Z	0.31	0.33	0.34	0.37
	NFE	11169	26599	33641	72047
Pruning	$Z(G_c)/Z(G)$	0.91	0.94	0.94	0.97
	$NFE(G_c)/NFE(G)$	1.27	3.39	3.28	6.38
Node consolidation	$Z(G_c)/Z(G)$	0.06	0.17	0.20	0.58
	$NFE(G_c)/NFE(G)$	0.90	2.84	3.26	6.70
Heuristic - type	$Z(G_c)/Z(G)$	0.84	0.93	0.92	0.99
	$NFE(G_c)/NFE(G)$	1.05	1.69	2.36	7.80
Heuristic - betweenness	$Z(G_c)/Z(G)$	0.89	0.93	0.94	1.07
	$NFE(G_c)/NFE(G)$	0.75	1.62	1.98	6.13
On-the-fly	$Z(G_c)/Z(G)$	0.89	0.96	0.96	1.03
	$NFE(G_c)/NFE(G)$	0.32	1.96	2.51	7.57

B

Appendix for Chapter 4: Network features

Table B.1 shows the distribution of features on the case study networks used in Chapter 4. The considered features are cameras, traffic lights, roundabouts, tunnels, and bridges.

Table B.1: Case study road networks used in this study. Incident locations are marked in orange; escape nodes in red, and police starting positions in blue.

Utrecht	Winterswijk	Manhattan
		

C

APPENDIX FOR CHAPTER 5: ADDITIONAL ANALYSES

C.1. UNSCALED SOLUTION QUALITY

Regardless of the generative model, the probability of interception increases with an increasing number of sensors for PRO and DPS. This shows that adaptive methods can utilize the information to improve the solutions. There is one problem instance for each combination of number of police units and number of sensors, where the adaptive methods (PTO and DPS) find a solution with a 100% probability of interception, which the non-adaptive techniques do not find. This solution is highly dependent on sensor information and preserving flexibility. With 10 police units on a 10x10 grid, achieving a probability of interception of close to 100% is possible, especially for the random walk model.

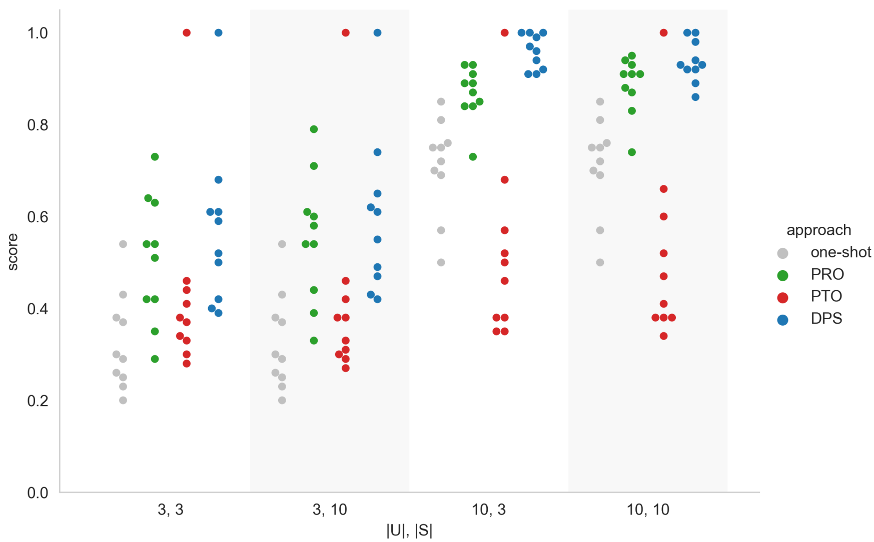


Figure C.1: Comparison of (unscaled) solution quality on the grid network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

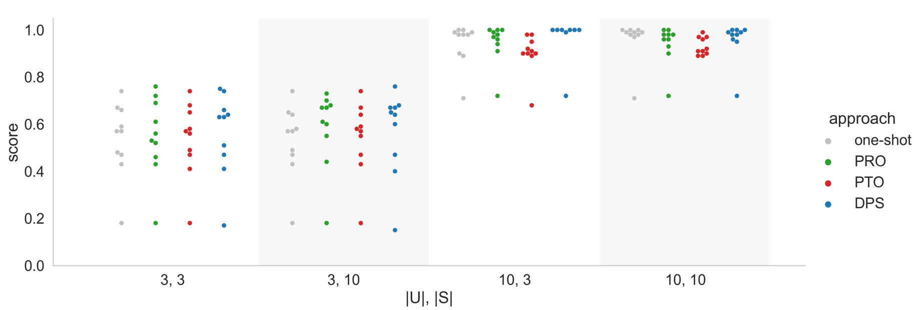


Figure C.2: Comparison of (unscaled) solution quality on the Manhattan road network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

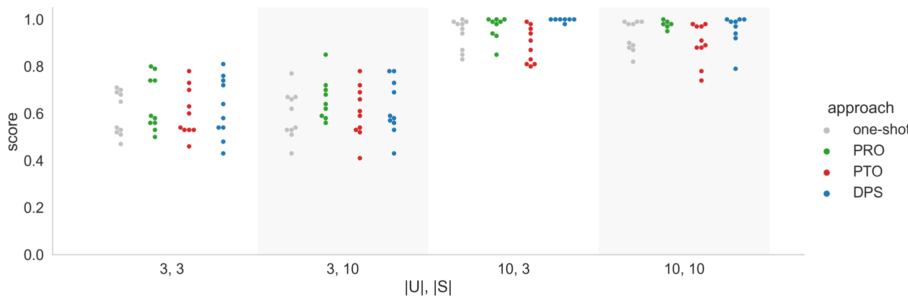


Figure C.3: Comparison of (unscaled) solution quality on the Utrecht road network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

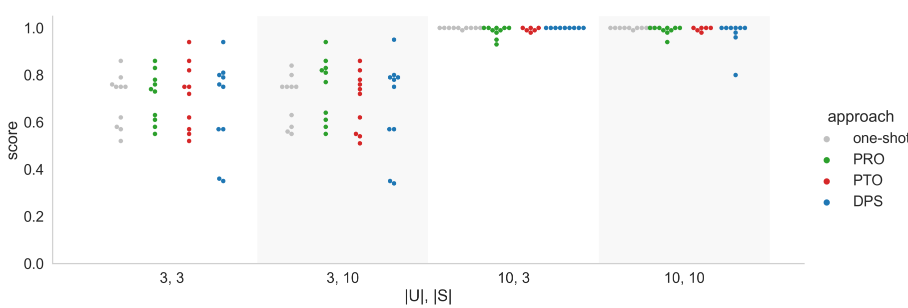


Figure C.4: Comparison of (unscaled) solution quality on the Winterswijk road network across problem instances for four solution approaches: one-shot optimization, periodic re-optimization (PRO), policy tree optimization (PTO), and direct policy search (DPS).

C.2. ANALYSIS OF CONVERGENCE

C.2.1. PRO

Figures C.5, C.6, C.7, C.8 show the results of the experiments where the total number of function evaluations used by PRO is 10 000. In the experiments presented in the main text (Figures 5.6, 5.7, 5.8, 5.9), PRO gets 10 000 function evaluations for each re-optimization, which occurs 3-7 times on average, depending on the problem instance, resulting in 30 000 - 70 000 function evaluations in total. These experiments show that, with a more limited number of function evaluations, PRO's performance decreases.

For the grid network, limiting the number of function evaluations only slightly decreases the solution quality: PRO still outperforms DPS and PTO for problem instances with three police units (Figure C.5). For the Manhattan network, the quality of the solutions found by PRO decreases significantly for most problem instances (Figure C.6). 2 000 - 3 000 function evaluations are often insufficient to converge, leading to poor solutions and lock-in effects. Across problem instances, the solution quality is on par or worse than the solutions found by DPS. For the Utrecht network, limiting the number of function evaluations for PRO decreases the quality of the solutions significantly for most problem instances (Figure C.7). On average, the quality of the solutions found by PRO is lower than the solutions found by one-shot optimization, PTO, and DPS. Due to the complexity of the Utrecht road network, many re-optimizations are triggered, reducing the number of function evaluations to 1 500 - 2 500, which is too few to reliably find good solutions. In contrast, for the Winterswijk network, the solution quality only slightly decreases: PRO still outperforms DPS and PTO for problem instances with three police units (Figure C.7).

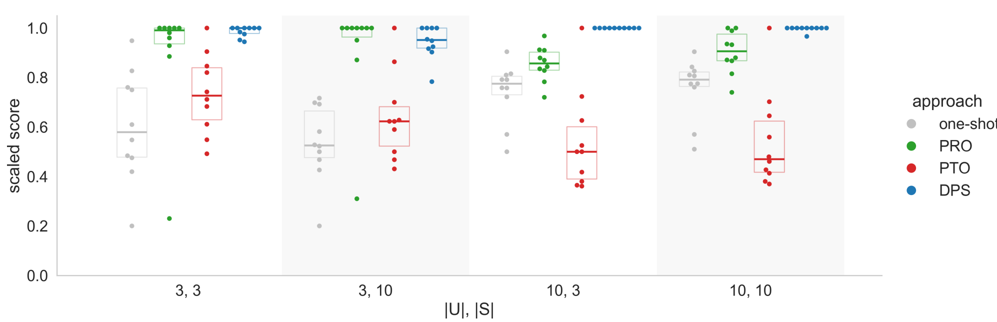


Figure C.5: Comparison of solution quality on the grid network, where the total nfe used by PRO is 10 000.

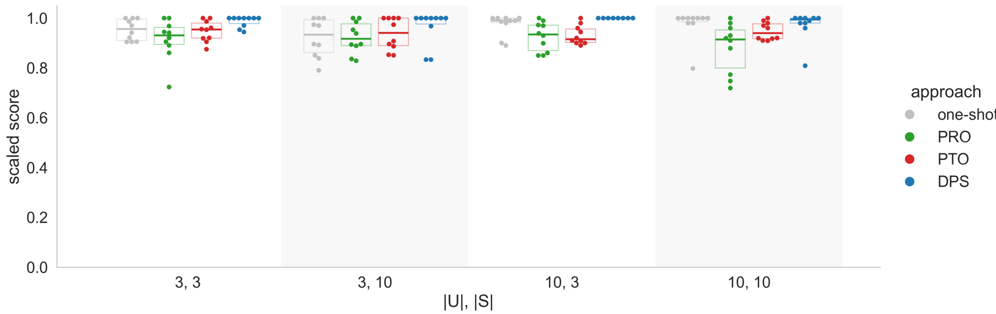


Figure C.6: Comparison of solution quality on the Manhattan road network, where the total nfe used by PRO is 10 000.

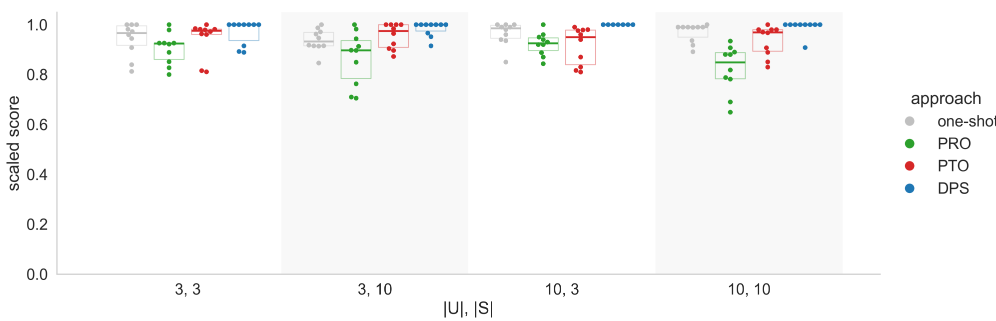


Figure C.7: Comparison of solution quality on the Utrecht road network, where the total nfe used by PRO is 10 000.

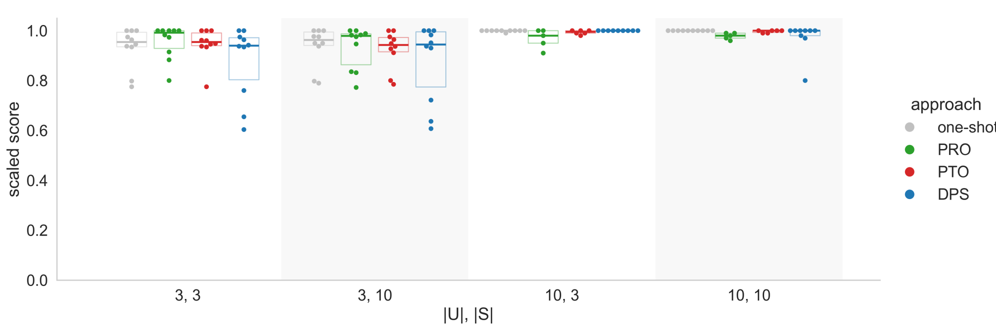


Figure C.8: Comparison of solution quality on the Winterswijk road network, where the total nfe used by PRO is 10 000.

Table C.1: Comparison of the solutions found by DPS, given 10 000 and 100 000 function evaluations.

network	$\|U\|$	$\|S\|$	instance	unscaled solution quality 10 000 nfe	100 000 nfe	best-found
Manhattan	3	3	4	0.15	0.15	0.18
	3	10	3	0.41	0.41	0.43
	3	10	4	0.40	0.42	0.48
	10	10	1	0.95	0.98	0.99
Utrecht	3	10	4	0.43	0.49	0.56
	3	10	6	0.56	0.56	0.68
Winterswijk	3	3	5	0.35	0.35	0.58
	3	3	6	0.57	0.57	0.75
	3	10	5	0.34	0.35	0.58
	3	10	6	0.57	0.57	0.81
	3	10	8	0.35	0.35	0.55

C.2.2. DPS

Table C.1 shows the solution quality found by DPS after 10 000 and 100 000 function evaluations for problem instances where DPS performed relatively poorly. The solution quality barely increases with an increased number of function evaluations. This demonstrates that the performance of DPS is not hindered by the convergence within the limited time frame.

C.2.3. PTO

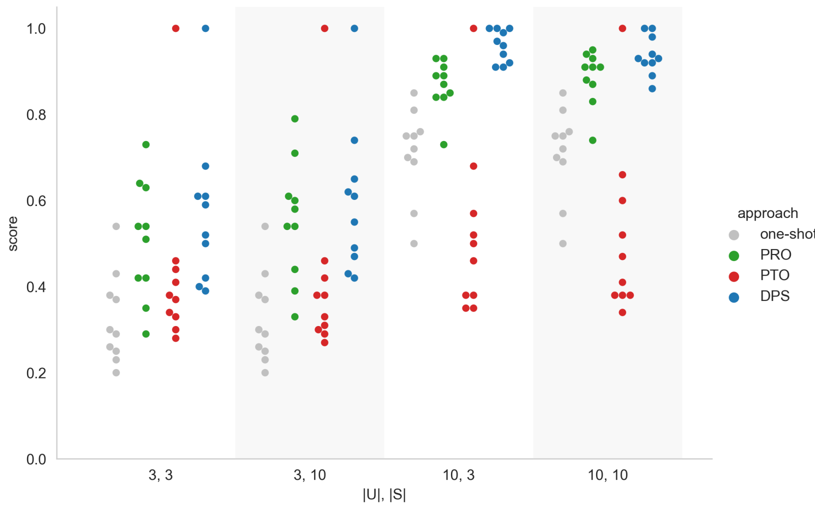
Table C.2 shows the solution quality found by PTO after 10 000 and 100 000 function evaluations for problem instances where PTO performed relatively poorly. The solution quality increases for all instances, sometimes even to 99% of the best-found solution quality. This demonstrates that the performance of PTO is primarily hindered by the convergence within the limited time frame. However, even with 100 000 function evaluations and 10 seeds, PTO is not able to find the best-found solution across approaches. The evolution process rarely produces feasible, well-performing policy trees. Therefore, PTO quickly converges to and rarely escapes from a local optimum, making it sensitive to the initial sample.

C.3. RANDOM WALKS

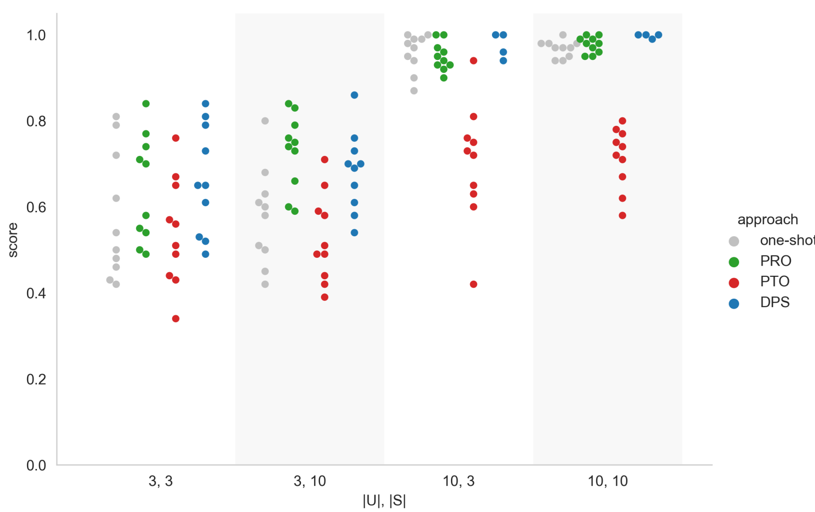
The number of intercepted routes is significantly higher when optimizing for a random walk model of the fugitive compared to a shortest path model. Comparatively, the random walk model generates routes more concentrated around the starting position of the fugitive, resulting in relatively easy interception. The shortest path model with 2% noise (1) results in relatively shorter routes, constraining the time that the police have for interception, and (2) results in routes that are less concentrated and hence it is more difficult to intercept a high percentage with limited number of police units.

Table C.2: Comparison of the solutions found by PTO, given 10 000 and 100 000 function evaluations.

network	$\|U\|$	$\|S\|$	instance	Unscaled solution quality 10 000 nfe	100 000 nfe	best-found
grid	3	3	0	0.33	0.33	0.39
	3	10	1	0.31	0.41	0.71
	10	3	2	0.68	0.73	0.94
	10	10	3	0.38	0.44	0.92
Manhattan	3	10	6	0.57	0.64	0.67
	10	3	7	0.86	0.92	1.0
	10	10	5	0.9	0.99	1.0
Utrecht	3	3	7	0.6	0.72	0.74
	3	10	7	0.64	0.72	0.73
	10	3	7	0.78	0.89	1.0
	10	10	7	0.8	0.85	1.0



(a) Shortest path with 2% noise



(b) Random walk

Figure C.9: Comparison of online optimization approaches, using an (a) shortest path with noise, and (b) random walk generative models for the fugitive.

ACKNOWLEDGEMENTS

This dissertation is the result of four years of learning, collaboration, and support from those around me. I am grateful to everyone who has contributed to this work and supported me – the following pages are for you.

First, I would like to thank my promotors, Alexander Verbraeck and Jan Kwakkel. Alexander, your support over the past few years—especially in the final months—has meant a lot to me. You encouraged me to be independent while always being available for feedback, discussions, and advice. You also showed me how rewarding it is to supervise and coach students, something I will take with me throughout my career. Jan, there seems to be no research topic you don't know something about. I've really appreciated your pragmatic approach, especially when I was making things unnecessarily complicated. You helped me structure my thoughts and improve my writing—and I still have a long list of books and podcasts to catch up on thanks to you. I would also like to thank Jelte Mense, my external advisor at the police, for helping me navigate the organization and embed its way of working into my research. You always took the time to check in before talking about work, which I really appreciated. And after three years of searching for the right data, you made sure I finally got access, leading to an extra chapter that became the cherry on top of this dissertation.

I also want to thank the committee members for their interest and support. First, Shane Henderson, for his detailed and constructive feedback and for his kindness during my research visit to Cornell University. Whether it was after a research meeting, a beer, or a bouldering session, I always left with a fresh perspective on my work. I would like to thank Simon Taylor for his kind words and warm welcome on our research visit to Brunel University and during the Winter Simulation Conference. Next, I would like to thank Natalie van der Wal for her kind feedback and for the enjoyable experience of co-supervising two master's students together. Her fresh perspective on behavior and decision-making research added depth and meaning to the work. Finally, I would like to thank Martijn Warnier for the constructive contributions, for checking in during late nights at TPM, and for the gezelligheid at lunch and TB cafe.

This dissertation also marks the end of my work with the National Police AI Lab. I would like to thank Theo van der Plas, Ron Boelsma, Marius Kok, and Bas Testerink for their support. TWO – Nieuwgein, the NU, and, long ago, the villa – made this research something larger than just a dissertation. In particular, I want to thank Tamar Kastelein for laying the foundation and being a great colleague. Thank you for believing in this research. I also want to thank everyone at the police who introduced me to the organization, the control room, and took me on shifts. In particular, Wout Boevé, for thinking big and pushing for this research to happen.

My PhD journey would have been much less enjoyable without my wonderful colleagues at TPM. A special thanks to the Policy Analysis PhDs (the 'pubquiz team'), particularly Alessandro, Ashta, Brayton, Damla, Ignasi, Jin, Joos, Liz, Nely, Omar, Palok, Ruth, Thorid, Zuzanna, and everyone else, for the endless coffees and good times. To the police-babies, Coen and Tess: even though we only shared the office for a few months, it was a pleasure. And to Jerico and Josephine, thank you for the heartfelt conversations.

I also want to thank my master's students— Rogier, Lavinia, Tom, Willemijn, Cécile, and Veerle. Your ideas added so much to the research, and working with you taught me a lot about myself and about supervising.

I am also grateful to my Very Important Paranymphs: Karoline and Isabelle. Karoline, from co-working in Mérida to our coffee-fueled walks in the final weeks, your support has meant so much. Your care for the world and the people around you is inspiring. Isabelle, I have learned so much from you. You were my foremost role-model, cheerleader and confidante. People say a PhD is lonely, but thanks to you, that was never the case. It amazes me that we have yet to run out of things to talk about. The secret to happiness? Always having a trip to look forward to. And cheers to never having to use the HPC again!

I would like to thank all my friends for keeping me grounded - in particular, Ilse, Roos, Laura, Katerina, Eline, and Lindy. You inspire me, and I am so proud of each of you.

I want to thank my family for your support, curiosity, and love. You have been with me since day 0. The older I get, the more I see how much I take after you — and I am proud of that. Finally, I'd like to thank my bib, Adrian, for always being there for me, for listening to my concerns and doubts, and for helping me see the bigger picture. Your whirlwind of energy has brought so much joy into my life. I am so happy to have you by my side.

Irene Sophia van Droffelaar
Amsterdam, March 2025

About the author

Irene van Droffelaar obtained her bachelor's degree in Chemistry from the University of Groningen. Inspired by a year as a student representative on the executive board of the Faculty of Science and Engineering, she transitioned to applied research, pursuing a master's degree in Engineering and Policy Analysis at Delft University of Technology. She graduated in 2020 with a thesis proposing a novel sampling approach for scenario discovery, a quantitative model-based method for scenario development. Afterward, she started a PhD trajectory in the Policy Analysis section of Delft University of Technology, collaborating with the Dutch Police through the ICAI National Police AI Lab. Her research focused on developing methods for decision support to the police control room to increase red-handed arrests.

Peer-reviewed publications

Published

1. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). Simulation–optimization configurations for real-time decision-making in fugitive interception. *Simulation Modelling Practice and Theory, 133*, 102923.

2. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2025). Graph Coarsening for Fugitive Interception. *Applied Network Science, 10*, 2.

Accepted

1. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). The effect of models of criminal behavior on police interception strategies. *Advances in Social Simulation: Proceedings of the 19th Social Simulation Conference.*

Under review

1. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). Timely Adaptive Strategies for Fugitive Interception.

Presentations

1. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). Optimizing interception: The use of computer simulation and optimization to (re)construct systems with uncertainty. *Police Academy*. Apeldoorn, The Netherlands.

2. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2024). The effect of models of fugitive behavior on police interception strategies. *Social Simulation Conference.* Kraków, Poland.

3. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2023). Timely adaptive strategies for police-fugitive interception. *Police AI Lab meeting.* Delft, The Netherlands. (poster)

4. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2023). Timely adaptive strategies for police-fugitive interception. *Annual Meeting of the Society for Decision Making Under Deep Uncertainty.* Delft, The Netherlands. (poster)

5. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2023). Optimizing interception: The use of computer simulation and optimization to (re)construct systems with uncertainty. *Police Academy.* Apeldoorn, The Netherlands.

6. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2022). Simulation-optimization configurations for fugitive interception. *Forum on interactive multiobjective optimization.* Jyväskylä, Finland

7. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2022). Simulation-optimization configurations for fugitive interception. *Winter Simulation Conference.* Singapore, Singapore. (poster) (best poster award)

8. Van Droffelaar, I.S., Van Schilt, I.M., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2022). Simulation. *Police AI Lab meeting.* Utrecht, The Netherlands.

9. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2022). Policy tree optimization for real-time decision-making under deep uncertainty. *Annual Meeting of the Society for Decision Making Under Deep Uncertainty.* Mexico City, Mexico.

10. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2022). Online optimization approaches for fugitive interception. *Conference of the Netherlands Research School on Transport, Infrastructure and Logistics.* Utrecht, The Netherlands. (best presentation award)

11. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2021). Applying the Flow Interception Problem to a fugitive situation. *International Conference on Computational Logistics.* Enschede, The Netherlands (online).

12. Van Droffelaar, I.S., Kwakkel, J.H., Mense, J. P. & Verbraeck, A. (2021). Optimizing interception. *Conference of the Netherlands Research School on Transport, Infrastructure and Logistics.* (online) (best presentation award)

best poster or presentation award

SUPERVISED MASTER'S THESES

1. Zuurdeeg, V.E.M. (2024) *Optimizing fugitive interception: A comparative study into the added value of including more realistic traffic conditions in fugitive interception models* [Master's thesis, Engineering and Policy Analysis, Delft University of Technology].

2. Van Heukelom, C.M.V. (2024) *Police Strategies for Fugitive Interception: A case study in the metro network of Rotterdam* [Master's thesis, Engineering and Policy Analysis, Delft University of Technology].

3. Tutuarima, W.A. (2023). *Criminal Fugitive Escape Routes: The influence of behavioral route-choice factors on criminal fugitive escape routes* [Master's thesis, Engineering, and Policy Analysis, Delft University of Technology].

4. Paoletti, L. (2022). *The role of DSSs in decision-making processes characterized by time pressure, uncertainty, and dynamism: an agent-based modeling approach* [Master's thesis, Complex Systems Engineering and Management, Delft University of Technology].

5. Kempenaar, P.T. (2022). *Prospective Criminal Escape Routes: An Exploration of Fugitive Escape Route Decision-Making using a Dual-Process Approach* [Master's thesis, Engineering and Policy Analysis, Delft University of Technology].

6. Van der Weerd, R. (2022) *Predicting plausible escape routes using reinforcement learning and graph representations.* [Master's thesis, MSc. Artificial Intelligence, University of Amsterdam]

TRAIL THESIS SERIES

The following list contains the most recent dissertations in the TRAIL Thesis Series. For a complete overview of more than 400 titles, see the TRAIL website: www.rsTRAIL.nl.

The TRAIL Thesis Series is a series of the Netherlands TRAIL Research School on transport, infrastructure and logistics.

Droffelaar, I.S. van, *Simulation-optimization for Fugitive Interception*, T2025/5, May 2025, TRAIL Thesis Series, the Netherlands

Fan, Q., *Fleet Management Optimisation for Ride-hailing Services: from mixed traffic to fully automated environments*, T2025/4, April 2025, TRAIL Thesis Series, the Netherlands

Hagen, L. van der, *Machine Learning for Time Slot Management in Grocery Delivery*, T2025/3, March 2025, TRAIL Thesis Series, the Netherlands

Schilt, I.M. van, *Reconstructing Illicit Supply Chains with Sparse Data: a simulation approach*, T2025/2, January 2025, TRAIL Thesis Series, the Netherlands

Ruijter, A.J.F. de, *Two-Sided Dynamics in Ridesourcing Markets*, T2025/1, January 2025, TRAIL Thesis Series, the Netherlands

Fang, P., *Development of an Effective Modelling Method for the Local Mechanical Analysis of Submarine Power Cables*, T2024/17, December 2024, TRAIL Thesis Series, the Netherlands

Zattoni Scroccaro, P., *Inverse Optimization Theory and Applications to Routing Problems*, T2024/16, October 2024, TRAIL Thesis Series, the Netherlands

Kapousizis, G., *Smart Connected Bicycles: User acceptance and experience, willingness to pay and road safety implications*, T2024/15, November 2024, TRAIL Thesis Series, the Netherlands

Lyu, X., *Collaboration for Resilient and Decarbonized Maritime and Port Operations*, T2024/14, November 2024, TRAIL Thesis Series, the Netherlands

Nicolet, A., *Choice-Driven Methods for Decision-Making in Intermodal Transport: Behavioral heterogeneity and supply-demand interactions*, T2024/13, November 2024, TRAIL Thesis Series, the Netherlands

Kougiatsos, N., *Safe and Resilient Control for Marine Power and Propulsion Plants*, T2024/12, November 2024, TRAIL Thesis Series, the Netherlands

Uijtdewilligen, T., *Road Safety of Cyclists in Dutch Cities*, T2024/11, November 2024, TRAIL Thesis Series, the Netherlands

Liu, X., *Distributed and Learning-based Model Predictive Control for Urban Rail Transit Networks*, T2024/10, October 2024, TRAIL Thesis Series, the Netherlands

Clercq, G. K. de, *On the Mobility Effects of Future Transport Modes*, T2024/9, October 2024, TRAIL Thesis Series, the Netherlands

Dreischerf, A.J., *From Caveats to Catalyst: Accelerating urban freight transport sustainability through public initiatives*, T2024/8, September 2024, TRAIL Thesis Series, the Netherlands

Zohoori, B., *Model-based Risk Analysis of Supply Chains for Supporting Resilience*, T2024/7, October 2024, TRAIL Thesis Series, the Netherlands

Poelman, M.C., *Predictive Traffic Signal Control under Uncertainty: Analyzing and Reducing the Impact of Prediction Errors*, T2024/6, October 2024, TRAIL Thesis Series, the Netherlands

Berge, S.H., *Cycling in the age of automation: Enhancing cyclist interaction with automated vehicles through human-machine interfaces*, T2024/5, September 2024, TRAIL Thesis Series, the Netherlands

Wu, K., *Decision-Making and Coordination in Green Supply Chains with Asymmetric Information*, T2024/4, July 2024, TRAIL Thesis Series, the Netherlands

Wijnen, W., *Road Safety and Welfare*, T2024/3, May 2024, TRAIL Thesis Series, the Netherlands

Caiati, V., *Understanding and Modelling Individual Preferences for Mobility as a Service*, T2024/2, March 2024, TRAIL Thesis Series, the Netherlands

Vos, J., *Drivers' Behaviour on Freeway Curve Approach*, T2024/1, February 2024, TRAIL Thesis Series, the Netherlands

Geržinič, N., *The Impact of Public Transport Disruptors on Travel Behaviour*, T2023/20, December 2023, TRAIL Thesis Series, the Netherlands

Dubey, S., *A Flexible Behavioral Framework to Model Mobility-on-Demand Service Choice Preference*, T2023/19, November 2023, TRAIL Thesis Series, the Netherlands

Sharma, S., *On-trip Behavior of Truck Drivers on Freeways: New mathematical models and control methods*, T2023/18, October 2023, TRAIL Thesis Series, the Netherlands

Ashkrof, P., *Supply-side Behavioural Dynamics and Operations of Ride-sourcing Platforms*, T2023/17, October 2023, TRAIL Thesis Series, the Netherlands

Sun, D., *Multi-level and Learning-based Model Predictive Control for Traffic Management*, T2023/16, October 2023, TRAIL Thesis Series, the Netherlands

Brederode, L.J.N., *Incorporating Congestion Phenomena into Large Scale Strategic Transport Model Systems*, T2023/15, October 2023, TRAIL Thesis Series, the Netherlands

Summary

Fugitive interception is a challenging task, requiring the police control room to decide in a matter of minutes on the optimal positions of police units to intercept a fleeing suspect. Models can support the police by suggesting interception positions. This dissertation addresses four barriers to using models for decision support in fugitive interception: timely simulation-optimization, effective representation of the search space, simulation of fugitive behavior, and adaptation to information updates.

About the Author

Irene S. van Droffelaar holds an MSc in Engineering and Policy Analysis from Delft University and Technology and a BSc in Chemistry from the University of Groningen. She conducted her PhD at the Faculty of Technology, Policy and Management of Delft University of Technology, as part of the National Police Artificial Intelligence Lab (NPAI).

TRAIL Research School ISBN 978-90-5584-360-2