Erik Le Blansch

Development of a Recipe-based Strategy for Scrap Sorting

Delft University of Technology

TU Delft

Development of a Recipe-based Strategy for Scrap Sorting

by

Erik Le Blansch

in partial fulfilment of the requirements for the degree of

Master of Science
in Mechanical Engineering

at the Department Maritime and Transport Technology of the Faculty of Mechanical Engineering of Delft University of Technology,

to be defended publicly on Tuesday June 29, 2026 at 15:00 AM

Student number:	4350804	
MSc track:	Multi-Machine Engineering	
Report number:	2026.MME.9185	
Supervisors:	Prof. dr. ir. D.L. Schott,	Responsible Supervisor
	Dr. ir. Y. Pang,	Supervisor
Thesis committee:	Prof. dr. ir. D.L. Schott,	TU Delft committee Chair, ME-MTT-MME
	Dr. ir. Y. Pang,	TU Delft committee member, ME-MTT-TEL
	Dr. ir. S.E. Offerman,	TU Delft committee member, ME-MSE-OFFERM
Date:	May 29, 2026	

Cover: Industrial steel furnace (2019) by Charlota Blunarova, Unsplash

TU Delft

Preface

This thesis marks the final step in completing my Master of Science in Mechanical Engineering at Delft University of Technology. The work presented in this report has been both challenging and rewarding, combining topics from steel recycling, sensor-based sorting, data processing, and algorithmic decision-making. It allowed me to explore how smart logistics can contribute to the more efficient use of recycled materials and the extraction of greater value from them.

When I was first offered the topic of scrap sorting, it was still a very broad and open-ended subject. This made the start of the project both interesting and challenging. A significant part of the work was to shape this broad theme into a clear, feasible, and meaningful thesis topic. Finding an approach that was realistic within the scope of a master thesis, while still contributing to the current state of knowledge and practice, became an important part of the process. Looking back, this challenge has made the project especially valuable to me, as it required not only technical work, but also careful framing, selection, and refinement of the research direction.

I would like to express my gratitude to my supervisors, Prof. dr. ir. D.L. Schott and Dr. ir. Y. Pang, for their guidance, feedback, and support throughout this thesis project. Their critical questions and suggestions helped me to improve the structure, technical depth, and clarity of this research. I especially appreciated their ability to provide direction while still allowing me the freedom to develop and refine my own approach.

I would also like to thank the faculty coordinators for their support in helping me resume my master programme after a period of gaining professional experience as an engineer. Their assistance made it possible for me to pick up the programme again and work towards completing this final step.

Furthermore, I would like to thank the members of the thesis committee for their time and for evaluating this work. In addition, I am grateful to the staff and fellow students at the Department of Maritime and Transport Technology for creating an inspiring academic environment.

Finally, I would like to thank my parents, family and friends for their support, patience, and encouragement during the thesis process. Their interest and motivation helped me to stay focused, especially during the more demanding stages of the project.

Erik Le Blansch
Delft, June 2026

Summary

English – *Steel production is essential for modern construction and manufacturing, but it is also resource-intensive and responsible for a substantial share of global CO_2 emissions. Increasing the use of recycled steel scrap is therefore an important route toward more circular and lower-emission steel production. However, scrap streams often contain individual pieces with highly variable chemical compositions. When these compositions are not controlled, residual and alloying elements can exceed the limits of high-grade steel recipes, causing scrap to be down-cycled into lower-value products or diluted with primary raw materials.*

Recent developments in sensor-based sorting technologies, such as LIBS, XRF, and XRT, make it increasingly possible to determine the material type, mass, and elemental composition of individual scrap pieces. Yet, this detailed piece-level information is not fully used in conventional sorting systems, which often classify scrap into broad material categories. This thesis addresses that gap by developing and simulating a recipe-based scrap sorting decision layer that allocates individual scrap items to steel recipes according to their chemical compatibility and economic value.

The proposed system uses a buffer of scanned scrap items and evaluates combinations of these items against chemical constraints derived from European Steel Standards. A scrap heap is considered feasible when the mass-weighted average composition of all selected items satisfies the lower and upper elemental limits of a target steel recipe, while also respecting a maximum heap capacity. The allocation problem is highly combinatorial, because many scrap items must be evaluated against many possible recipes. Therefore, several heuristic algorithms were developed and compared, including single-run greedy, multi-pass greedy, multi-start greedy, and variants with local search.

The simulation framework was implemented using synthetically generated scrap data based on steel composition ranges, because large-scale public datasets with piece-level scrap compositions are not available. System performance was evaluated using item conversion, mass conversion, heap utilisation, and economic value. These indicators were used to compare algorithmic performance and to assess the effect of key system parameters, such as buffer size, heap capacity, and the number of available heaps.

The results show that greedy heuristic methods are suitable for the proposed sorting decision layer. In particular, the multi-pass greedy algorithm provided the best balance between solution quality and computational efficiency. Multi-start and local-search variants achieved only limited additional performance improvements while requiring substantially more computation time. The buffer size analysis showed that performance improves strongly at small buffer sizes, but that gains diminish beyond approximately 700-1000 scrap items. A buffer size of 1000 items was therefore selected as a stable baseline.

Further simulation experiments showed a clear trade-off between mass conversion, heap utilisation, and economic value. Increasing heap capacity and the number of heaps generally improves mass conversion and value creation, but can reduce heap utilisation because larger heaps become harder to fill completely within chemical constraints. A balanced operating region was found around seven heaps with a heap capacity of 12000 kg, achieving approximately 78% mass conversion and 89% heap utilisation while maintaining strong economic performance.

Overall, this thesis demonstrates that recipe-based scrap sorting is a promising strategy for increasing the value recovered from steel scrap. By combining sensor-based item characterisation, recipe constraints, and scalable allocation algorithms, scrap sorting can shift from broad classification toward value-oriented recipe allocation. Although the model relies on synthetic data and assumes accurate sensor measurements, it provides a practical blueprint for future decision-support systems in high-grade scrap-based steel production.

Nederlands – *Staalproductie is essentieel voor moderne bouw en productie, maar is ook grondstofintensief en verantwoordelijk voor een aanzienlijk deel van de wereldwijde CO_2-uitstoot. Het vergroten van het gebruik van gerecycled staalschroot is daarom een belangrijke route naar meer circulaire staalproductie met lagere emissies. Schrootstromen bestaan echter vaak uit afzonderlijke stukken met sterk variërende chemische samenstellingen. Wanneer deze samenstellingen niet worden gecontroleerd, kunnen residuele en legeringselementen de grenswaarden van hoogwaardige staalrecepten overschrijden. Hierdoor wordt schroot vaak gedowncycled naar producten met een lagere waarde of verdund met primaire grondstoffen.*

Recente ontwikkelingen in sensor-gebaseerde sorteertechnieken, zoals LIBS, XRF en XRT, maken het steeds beter mogelijk om het materiaaltype, de massa en de elementaire samenstelling van individuele schrootstukken te bepalen. Deze gedetailleerde informatie op stukniveau wordt echter nog niet volledig benut in conventionele sorteersystemen, die schroot vaak indelen in brede materiaalklassen. Deze thesis richt zich op deze leemte door een recept-gebaseerde beslislaag voor schrootsortering te ontwikkelen en te simuleren, waarmee individuele schrootstukken worden toegewezen aan staalrecepten op basis van chemische geschiktheid en economische waarde.

Het voorgestelde systeem gebruikt een buffer met gescande schrootstukken en beoordeelt combinaties van deze stukken aan de hand van chemische beperkingen uit Europese staalnormen. Een schroothoop wordt als haalbaar beschouwd wanneer de massa-gewogen gemiddelde samenstelling van alle geselecteerde stukken voldoet aan de onder- en bovengrenzen van de elementen binnen een doelrecept, terwijl ook de maximale hoopcapaciteit wordt gerespecteerd. Het toewijzingsprobleem is sterk combinatorisch, omdat veel schrootstukken moeten worden beoordeeld voor veel mogelijke recepten. Daarom zijn verschillende heuristische algoritmen ontwikkeld en vergeleken, waaronder single-run greedy, multi-pass greedy, multi-start greedy en varianten met local search.

Het simulatiekader is geïmplementeerd met synthetisch gegenereerde schrootdata op basis van staalcompositiebereiken, omdat grootschalige openbare datasets met schrootsamenstellingen op stukniveau niet beschikbaar zijn. De systeemprestaties zijn geëvalueerd met itemconversie, massaconversie, hoopbenutting, en economische waarde. Deze indicatoren zijn gebruikt om algoritmische prestaties te vergelijken en om het effect van belangrijke systeemparameters te beoordelen, zoals buffergrootte, hoopcapaciteit en het aantal beschikbare hopen.

De resultaten laten zien dat greedy heuristische methoden geschikt zijn voor de voorgestelde beslislaag. Met name het multi-pass greedy algoritme biedt de beste balans tussen oplossingskwaliteit en rekenefficiëntie. Multi-start- en local-search-varianten leveren slechts beperkte extra prestatieverbeteringen op, terwijl zij aanzienlijk meer rekentijd vereisen. De analyse van de buffergrootte laat zien dat de prestaties sterk toenemen bij kleine buffers, maar dat de meerwaarde afneemt vanaf ongeveer 700-1000 schrootstukken. Daarom is een buffergrootte van 1000 stukken gekozen als stabiele basisconfiguratie.

Verdere simulatie-experimenten laten een duidelijke afweging zien tussen massaconversie, hoopbenutting en economische waarde. Het vergroten van de hoopcapaciteit en het aantal hopen verbetert over het algemeen de massaconversie en waardecreatie, maar kan de hoopbenutting verlagen doordat grotere hopen moeilijker volledig te vullen zijn binnen de chemische beperkingen. Een gebalanceerd werkgebied werd gevonden rond zeven hopen met een hoopcapaciteit van 12000 kg, waarbij ongeveer 78% massaconversie en 89% hoopbenutting werd bereikt, terwijl sterke economische prestaties behouden bleven.

Deze thesis toont aan dat recept-gebaseerde schrootsortering een veelbelovende strategie is om meer waarde uit staalschroot terug te winnen. Door sensor-gebaseerde karakterisering op stukniveau, receptbeperkingen en schaalbare toewijzingsalgoritmen te combineren, kan schrootsortering verschuiven van brede classificatie naar waarde-gerichte recepttoewijzing. Hoewel het model gebaseerd is op synthetische data en uitgaat van nauwkeurige sensormetingen, biedt het een praktische basis voor toekomstige beslissingsondersteunende systemen voor hoogwaardige schroot-gebaseerde staalproductie.

AI Statement

For this ME-MME MSc Thesis ME54035, I have used Generative AI to improve the grammar, style, and/or spelling of the text. In all cases, I have reviewed and corrected the work and remain fully responsible for the content of the report.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

Abbreviation	Definition
CEV	Carbon Equivalent Value
DEM	Discrete Element Method
EAF	Electric Arc Furnace
EN	European Standard
EU	European Union
ILP	Integer Linear Programming
ISO	International Organisation for Standardisation
KPI	Key Performance Indicator
LIBS	Laser-induced Breakdown Spectroscopy
MILP	Mixed-Integer Linear Programming
MIS	Magnetic Induction Spectroscopy
NEN	Netherlands Standardization Institute
NIR	Near-Infrared Spectroscopy
RGB-D	Red-Green-Blue plus depth data
VBA	Visual Basic for Applications
XRF	X-ray Fluorescence
XRT	X-ray Transmission

Symbols

Symbol	Definition	Unit
Sets		
$\mathcal{E}$	Set of chemical elements considered in the model	[-]
$\mathcal{R}$	Set of available steel recipes	[-]
$\mathcal{S}$	Set of available scrap items in the buffer	[-]
$\mathcal{S}_{\mathrm{rem}}$	Set of remaining, unassigned scrap items in allocation round t	[-]
H_r	Set of scrap items assigned to the candidate heap for recipe r	[-]
$\hat{H}_t$	Set of scrap items assigned to the selected heap in allocation round t	[-]
$\tilde{H}_r$	Temporary heap configuration for recipe r used during multi-start or local-search procedures	[-]
Indices		
e	Index of chemical element	[-]
r	Index of steel recipe	[-]
i	Index of scrap item	[-]
j	Index of scrap item selected for removal or replacement	[-]
k	Index of candidate scrap item selected for insertion or replacement	[-]

Symbol	Definition	Unit
s	Index of random start in the multi-start greedy algorithm	[-]
t	Index of allocation round	[-]
iter	Index of local-search iteration	[-]
Parameters		
m_i	Mass of scrap item i	[kg]
$x_{i,e}$	Mass fraction of element e in scrap item i	[-]
$L_{r,e}$	Lower bound on the mass fraction of element e in recipe r	[-]
$U_{r,e}$	Upper bound on the mass fraction of element e in recipe r	[-]
M	Maximum allowable mass of a heap	[kg]
B	Buffer size, defined as the maximum number of scrap items available to the algorithm	[-]
T	Maximum number of heaps constructed in the allocation process	[-]
p_r	Market price of steel produced according to recipe r	[€/kg]
N	Target number of synthetic scrap items generated during data generation	[-]
N_{starts}	Number of random starting configurations in the multi-start greedy algorithm	[-]
$I_{\max}$	Maximum number of local-search iterations	[-]
$m_{\min}$	Minimum scrap item mass used during random mass generation	[kg]
$m_{\max}$	Maximum scrap item mass used during random mass generation	[kg]
Derived quantities		
W_r	Total mass of the candidate heap associated with recipe r	[kg]
$\hat{W}_t$	Total mass of the selected heap in allocation round t	[kg]
$\tilde{W}_r$	Temporary total heap mass used during multi-start or local-search procedures	[kg]
W'_r	Tentative total mass of candidate heap H_r after adding a candidate scrap item	[kg]
$A_{r,e}$	Total mass of element e contained in heap H_r	[kg]
$\tilde{A}_{r,e}$	Temporary total mass of element e used during multi-start or local-search procedures	[kg]
$A'_{r,e}$	Tentative total mass of element e in candidate heap H_r after adding a candidate scrap item	[kg]
$y_{r,e}$	Mass fraction of element e in heap H_r, defined as $A_{r,e}/W_r$	[-]
$\bar{x}'_{r,e}$	Tentative mass-weighted concentration of element e after adding a candidate scrap item	[-]
V_r	Economic value of the candidate heap constructed for recipe r	[€]
$\hat{V}_t$	Economic value of the selected heap in allocation round t	[€]
$\hat{r}_t$	Recipe corresponding to the selected heap in allocation round t	[-]
$N_{\mathrm{solutions}}$	Total number of candidate heap combinations	[-]

Symbol	Definition	Unit
Additional quantities		
$\mathcal{T}_1$	Input recipe table used for scrap data generation	[-]
$\mathcal{T}_2$	Candidate scrap table after sampling recipe rows with replacement	[-]
$\mathcal{T}_3$	Output table containing generated scrap items	[-]
$\mathcal{T}_{4,t}$	Candidate-heap result table for allocation round t	[-]
$\mathcal{T}_5$	Final allocation and annotation table	[-]
j_i	Randomly sampled recipe-row index used to generate scrap item i	[-]
$L_{i,e}$	Lower elemental bound used to generate the composition of sampled scrap item i	[-]
$U_{i,e}$	Upper elemental bound used to generate the composition of sampled scrap item i	[-]

1

Introduction

Steel is an essential material for modern infrastructure, transport, energy systems, and manufacturing, but its production remains highly resource- and emission-intensive. The World Steel Association [1] reported 1885 million tonnes of crude steel production in 2024, more than twice the 850 million tonnes produced in 2000. At the same time, steel production is responsible for approximately 8% of global CO_2 emissions, with primary steelmaking from iron ore accounting for the majority of these emissions [2]. Increasing the use and value of recycled steel scrap is therefore an important route towards reducing the dependence on virgin raw materials and making secondary steelmaking more competitive. In the EU, steel scrap already represents a substantial share of steel production, with the proportion of steel scrap used in relation to crude steel production reported at 56% [3]. However, using more scrap is not only a matter of quantity, but also of quality and allocation: scrap must be sorted and combined in such a way that it can satisfy the chemical requirements of valuable steel grades. This thesis therefore investigates how sensor-based composition data can be translated into recipe-based sorting decisions that increase the value recovered from scrap steel.

1.1. Background and motivation

One of the most effective strategies to reduce the environmental footprint of steel production is to increase the use of recycled steel scrap. Unlike many other materials, steel can be recycled repeatedly without losing its fundamental mechanical properties [1]. The use of steel scrap reduces the need for primary raw materials and can substantially lower energy consumption and CO_2 emissions compared with primary steel production [3, 4]. Therefore, scrap-based steel production plays an important role in the transition towards a more circular and low-carbon steel industry.

However, the value of scrap depends strongly on its chemical composition. Scrap streams originate from a wide range of products and end-of-life applications, which results in large compositional variation between individual scrap pieces [5]. Residual or tramp elements such as copper and tin, as well as alloying elements such as chromium when present in unsuitable steel grades, can limit the range of steel grades that can be produced from recycled material [4]. If these elements exceed the allowable composition limits, scrap may need to be diluted with primary raw materials or down-cycled into lower-value steel products [6].

The motivation of this thesis is therefore not only to increase the amount of steel scrap used, but also to increase the value recovered from it to make recycled steel economically more competitive with primary steelmaking from mined raw materials. Sensor-based scrap analysis offers the possibility to measure composition before melting, but this information must still be translated into sorting decisions. This creates the need for a recipe-based approach in which scrap pieces are combined so that their mass-weighted average composition satisfies the chemical limits of valuable steel grades.

1.2. Research gap

Recent advances in sensor-based scrap analysis have made it increasingly feasible to determine the elemental composition of individual scrap pieces. Handheld XRF and LIBS systems can support material characterisation in scrap-yard environments [7], while automated XRF and LIBS-based systems have shown potential for composition-based and alloy-level sorting of post-consumer metal scrap [8, 9]. These technologies can provide detailed information that could be used to improve the utilisation of scrap in steel production.

Despite these technological developments, the available compositional information is not yet fully exploited at the system level. Industrial scrap sorting commonly relies on broad material classes, origin, or standardised scrap categories, meaning that only partial information about the true scrap chemistry is retained [5]. Existing studies show the value of better compositional information and sensor-based classification, but they do not yet address how individual scrap pieces should be allocated to recipe-compliant heaps under mass, capacity, and value constraints.

A key limitation is therefore the absence of a smart “sorting decision layer” that converts piece-by-piece composition data into recipe-based allocation decisions. Since modern steel grades are defined by strict chemical composition boundaries, the final feasibility of a steel charge depends on the mass-weighted average composition of all selected scrap pieces. By developing an algorithm that searches for profitable and feasible combinations of scrap items, sensor data can be translated into practical sorting decisions that improve both material utilisation and economic value recovery.

1.3. Problem definition

Steel producers and scrap recycling companies increasingly rely on scrap metal as an input material for electric arc furnace (EAF) steel production [2]. While scrap recycling significantly reduces energy consumption and greenhouse gas emissions compared to primary steel production [3], it also introduces challenges related to the chemical variability of scrap materials [5]. Individual scrap pieces often contain different concentrations of alloying and tramp elements, which makes it difficult to control the final chemical composition of the produced steel [4].

Modern steel grades are defined by strict chemical composition boundaries according to standardised steel recipes. During steelmaking, the final composition of the melt is determined by the mass-weighted average of all input materials. When scrap pieces with unknown or incompatible compositions are combined, unwanted elements can accumulate and exceed the allowable limits for certain steel grades.

This phenomenon, commonly referred to as the build-up of tramp elements, reduces the range of steel grades that can be produced from recycled scrap and often forces producers to down-cycle scrap into lower-value steel products [6] or dilute impurities using primary raw materials [5].

Recent advances in sensor-based scrap analysis make it possible to measure the elemental composition of individual scrap pieces before they enter the steelmaking process [7]. This information could be used to combine scrap pieces in such a way that their weighted average composition satisfies the chemical boundaries of specific steel recipes. However, determining which combinations of scrap pieces form feasible and economically attractive steel recipes is a complex combinatorial problem. Large scrap streams contain thousands of individual items with different masses and chemical compositions, which have to be evaluated against hundreds of possible steel recipes.

The scientific challenge addressed in this thesis is therefore the development of a new algorithmic method that can efficiently allocate individual scrap items to steel recipes based on their chemical composition, while maximising economic value and making efficient use of available sorting capacity. By modelling and evaluating these allocation strategies in a simulation environment, this research aims to investigate how piece-by-piece scrap composition data can be used as a smart sorting decision layer to improve the performance of existing modern scrap sorting systems.

1.4. Research objective and main research question

To address this research gap, this thesis aims to develop a new sorting decision system that can sort (match) scrap pieces according to EU steel standards into the most profitable combination. This follows the main research question:

"How can a sensor-based scrap sorting system be developed and simulated to enable recipe-based steel production and maximise the value from large volumes of scrap metal?"

1.5. Sub-research questions

To answer the main research question, the research is devised in the following subsequent sub-research questions:

1. What is the state-of-the-art of sensor-based scrap metal scanning and sorting techniques, and to what extent do they meet the requirements for recipe-based steel production?
2. How can a system be developed that allocates scrap metal into sorted scrap streams based on the chemical composition of steel recipes according to European Steel Standards?
3. What kind of algorithms are suitable for assigning scrap items within the proposed sorting system, considering solution quality and computational efficiency?
4. How can the performance of the proposed scrap sorting system be quantitatively evaluated and tuned for maximal value creation and space efficiency?
5. How does the proposed scrap sorting system perform under varying system configurations?

1.6. Research approach

To address the main research question, this thesis begins with a literature review on the capabilities of state-of-the-art sensor-based scanning and sorting techniques (sub-research question 1). This provides the foundation for answering sub-research question 2, which focuses on the design of a sorting decision layer that allocates scrap based on the compositional data obtained from these scanning techniques.

Next, an appropriate algorithm is selected (sub-research question 3) to evaluate and implement the sorting decisions within this layer. Subsequently, key performance indicators are defined to assess and tune the system for maximum scrap value extraction (sub-research question 4).

Finally, a series of numerical experiments is conducted in line with sub-research question 5 to evaluate system performance under varying input parameters, such as the number of available heaps and the maximum capacity per heap.

1.7. Thesis Outline

The thesis is structured according to the research approach described in the Section 1.6. From this approach, a logical outline is visualised in Figure 1.1. This figure shows a Literature, Methodology, Results, Discussion, and Conclusion segment, each divided up into subsections that discuss the relevant subjects to answer the sub-research questions presented in Section 1.5.

The literature (Chapter 2) review addresses the first research question by analysing scrap separation techniques (Section 2.2), modern sensor-based sorting systems (Section 2.3), and sorting strategies (Section 2.4). It evaluates whether current state-of-the-art methods provide sufficiently accurate compositional data and feasible sorting speeds to support the proposed decision layer.

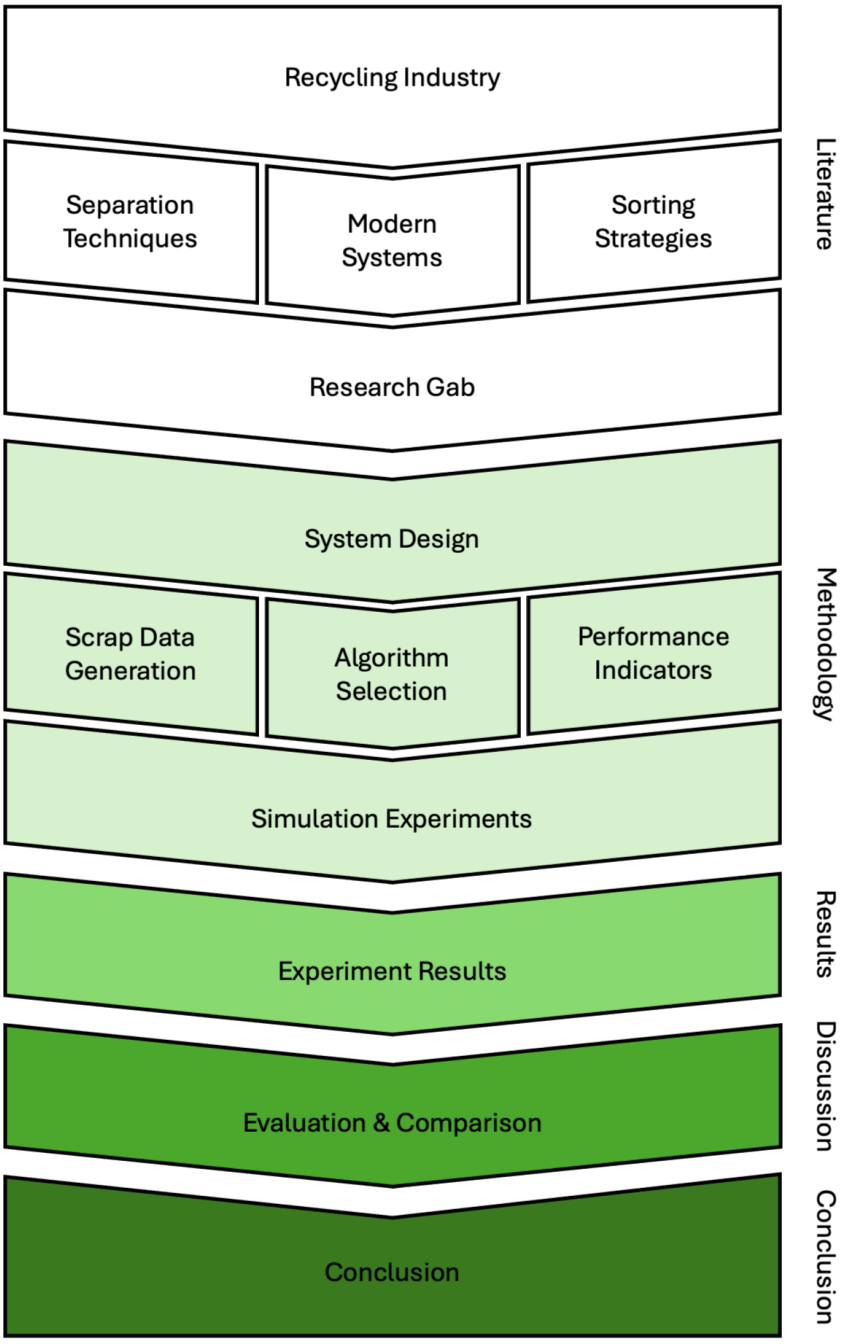


Figure 1.1: Flow diagram of the thesis outline based on Section 1.6 and Section 1.7.

The methodology part describes the development of the different components needed for the sorting decision layer, what algorithms are considered, and how the performance is evaluated in numerical simulation experiments. Thereby answering the second research question in Chapter 3, the third in Chapter 4, and the fourth in Chapter 5.

The results of the experiments described in Chapter 5 are presented in Chapter 6. The final sub-research question is addressed by analysing the sensitivity of the performance indicators to different parameter value combinations. In addition, supporting material is provided in the appendices: Appendix A contains the paper summary, Appendix B presents example Excel tables used in the simulation workflow, and Appendix C provides additional statistics to illustrate the structure of the solutions generated by the Sorting Decision Layer.

The discussion interprets the results presented in Chapter 7 by analysing the feasibility and scalability of the proposed system (Section 7.2), its limitations (Section 7.3), and its performance in comparison with existing scrap sorting methods (Section 7.4). The thesis concludes in Chapter 8 by summarising the answers to the five sub-research questions (Section 8.1), drawing conclusions with respect to the main research question (Section 8.2), and outlining recommendations for industrial implementation (Section 8.3) and future research (Section 8.4).

2

Literature Review

The introduction (Chapter 1) found that increasing the value of recycled steel scrap requires more than simply increasing the amount of scrap used in steel production. Because scrap pieces differ in chemical composition, the central challenge is to determine whether available sorting technologies can provide the information needed for recipe-based allocation. This literature review therefore forms the first step in the development of the proposed Sorting Decision Layer. It examines how scrap is currently processed, which sensing techniques can characterise individual scrap pieces, and whether existing sorting systems and optimisation approaches are sufficient to translate piece-level information into recipe-based steel production decisions.

This chapter answers the first sub-research question: "What is the state-of-the-art of sensor-based scrap metal scanning and sorting techniques, and to what extent do they meet the requirements for recipe-based steel production?" It first provides an overview of the steel recycling industry in Section 2.1, followed by detection techniques in Section 2.2, sensor-based sorting systems in Section 2.3, and existing approaches to automated scrap sorting in Section 2.4. The chapter concludes in Section 2.5 by summarising the literature findings and identifying the research gap addressed in this thesis.

2.1. Overview of steel recycling industry

Steel production relies heavily on recycled scrap metal as a raw material. Scrap originates from many sources, such as end-of-life vehicles, demolished buildings, industrial off-cuts, and consumer products [10]. One important advantage of steel is that it can be recycled repeatedly without substantial loss of properties if impurities are controlled. As a result, steel is the most recycled metal globally by mass, and recycled scrap accounts for about 30% of global steel production. In regions with high scrap availability, scrap-based production reaches 90%, while other regions still depend more on newly mined iron [11].

In the past, scrap sorting was a largely manual process. Early scrapyards depended on visual inspection, hand sorting, and simple magnetic separation to separate ferrous metals from other materials. As steel recycling volumes increased during the 20th century, mechanical processing became more common. Large shredders were introduced to process scrap streams like automobiles. During this period, scrap was classified by origin into home scrap, prompt (industrial) scrap, and obsolete scrap. Home and prompt scrap generally have known compositions and are easier to reuse, while obsolete scrap is more diverse and requires more sorting [12].

In modern steel production, scrap sorting is a major industrial operation. Steelmakers demand scrap with controlled chemical composition to meet quality requirements, especially for high-grade steels [13]. As a result, scrap processing facilities use a combination of physical, mechanical, and manual methods to increase scrap quality. Magnetic separation is still the primary technique for separating steel from mixed material streams, especially after shredding [14]. Eddy current separation is used to remove non-ferrous metals, improving the purity of ferrous scrap [15].

Despite automation, manual sorting is still used to remove visible contaminants or to separate specific scrap grades. However, manual methods are labour-intensive, inconsistent, and expose workers to safety risks [10]. This has stimulated the adoption of automated and semi-automated sorting technologies. Industrial facilities increasingly use handheld analysers and automated inspection systems to distinguish between carbon steel, alloy steel, and stainless steel [7]. These systems allow scrap to be grouped into standardised grades that meet steelmakers' specifications.

The economic aspect of sorting is very important. Well-sorted, low-impurity scrap offers higher market prices than mixed or contaminated scrap [6]. Thereby, effective sorting helps control residual elements such as copper and tin, which cannot be removed during the melting process and negatively affect steel properties [4]. Increased levels of these elements may cause surface cracking, reduced ductility, or limited formability [16]. Therefore, careful scrap separation is necessary, especially for high-strength steels with tight composition ranges.

2.2. Detection techniques

The previous section showed that steel recycling increasingly depends on reliable scrap quality control to limit residual elements and meet steelmakers' specifications. This section therefore reviews detection techniques used in automated sorting. The techniques are discussed from robust bulk pre-sorting methods, such as magnetic, eddy current, and induction-based detection, to density-, composition-, spectral-, and vision-based sensors. For each method, the principle, practical relevance, and limitations for recipe-based steel scrap sorting are highlighted, with attention to industrial implementation and performance.

Magnetic separation is widely used as an initial sorting step for shredded scrap. It exploits the magnetic force acting on magnetisable particles in a non-uniform magnetic field. As illustrated in Figure 2.1, a magnetic pulley deflects ferrous tramp iron from the normal trajectory of non-magnetic material. This makes the method robust, inexpensive, and suitable for high-throughput bulk pre-sorting. However, because it mainly separates magnetic from non-magnetic material, it cannot distinguish between different steel grades or provide alloy-level composition data [14].

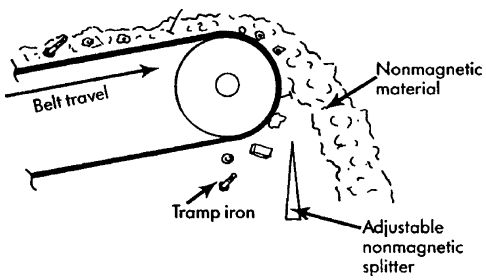


Figure 2.1: Magnetic pulley separation of tramp iron from non-magnetic materials [14].

Eddy current separation is commonly used after magnetic separation to recover non-ferrous metals from scrap streams. As shown in Figure 2.2, alternating magnetic poles induce currents in conductive particles, generating a repulsive force that throws metals such as aluminium or copper beyond the non-metal fraction. The method is fast and suitable for bulk pre-sorting, but separation depends strongly on particle size, shape, conductivity, and machine settings [15, 17].

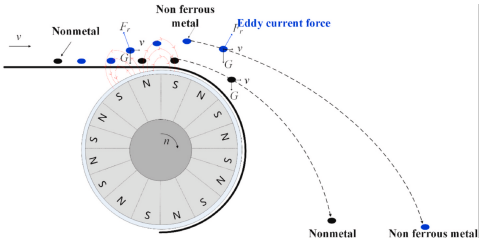


Figure 2.2: Eddy current separation [17].

Induction-based sorting uses electromagnetic coils to identify conductive scrap from its magnetic response. As shown in Figure 2.3, excitation coils generate an alternating field and receiving coils measure the response after interaction with the metal. The signal can support metal classification, especially when combined with vision, but it reflects conductivity and geometry rather than elemental composition and is therefore less suitable for precise alloy identification [18].

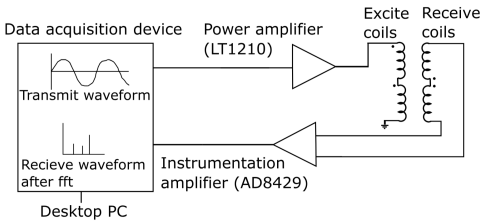


Figure 2.3: Schematic of an induction-based sensing system for scrap metal classification [18].

X-ray transmission (XRT) detection separates particles by measuring differences in X-ray absorption as they pass between an X-ray source and detector. As shown in Figure 2.4, the detected attenuation image can be classified and used to trigger physical separation, typically by air jets. Because absorption is influenced by density, thickness, and atomic density, XRT is effective when valuable particles show a clear contrast with the surrounding waste, such as dense cassiterite in lighter rock. In industrial applications, the technique can process large material streams continuously, enabling early rejection of unwanted fractions and reducing downstream processing costs and energy consumption. This makes XRT useful for high-throughput coarse pre-sorting, but it does not provide detailed elemental composition like XRF or LIBS. Therefore, its usefulness for recipe-based steel scrap classification is limited [19].

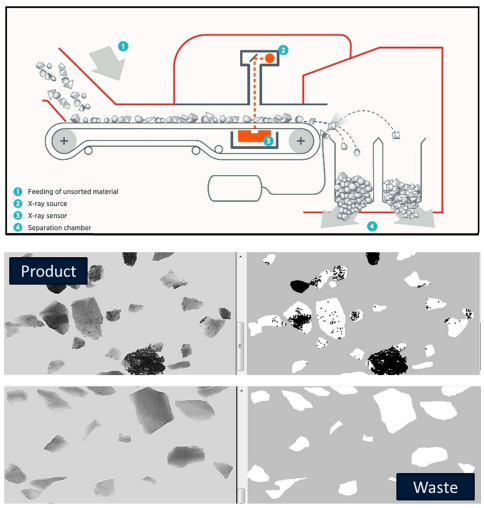


Figure 2.4: XRT sorting principle and example attenuation images [19].

X-ray fluorescence (XRF) sorting identifies scrap pieces from the characteristic secondary X-rays emitted after excitation by an X-ray source. As shown in Figure 2.5, particles are singulated on a vibratory feeder, scanned in the detection zone, classified by the processing unit, and separated by air jets. Because the measured spectrum is linked to elemental composition, XRF is suitable for alloy-specific sorting and quality control. However, industrial chute sorters must classify objects within milliseconds, so accuracy decreases when alloys have only small compositional differences. Light elements and contaminated surfaces can also be challenging, limiting use in high-speed recipe-based steel sorting [8].

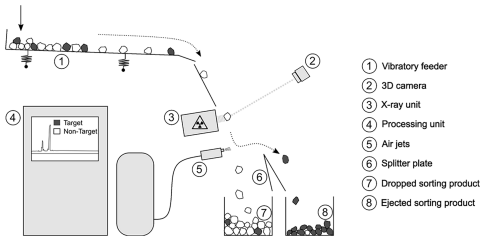


Figure 2.5: Operating principle of an XRF chute sorter for alloy-specific scrap sorting [8].

Laser-induced breakdown spectroscopy (LIBS) uses focused laser pulses to ablate a small area of the scrap surface, forming a plasma whose emitted spectrum contains element-specific atomic and ionic lines. As shown in Figure 2.6, the laser is directed onto the scrap piece, while the emitted light is collected by a spectrometer and processed into a corrected spectrum. This makes LIBS highly relevant for alloy-level sorting, especially because it can detect light elements more effectively than XRF. However, reliable classification requires stable spectra, surface preparation or compensation for contamination, and suitable preprocessing such as baseline correction to reduce spectral noise and improve measurement consistency. Although conventional machine-learning processing can be too slow, deep-learning approaches can classify many spectra simultaneously and support real-time industrial sorting [9].

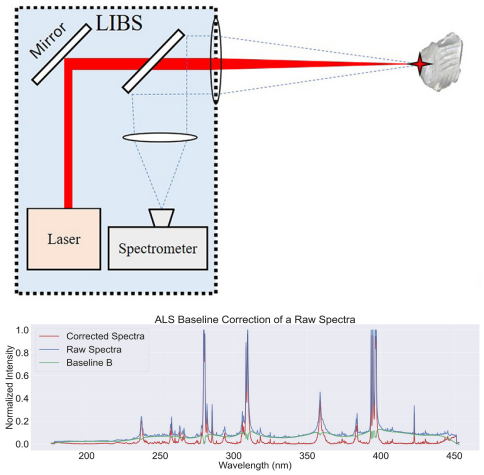


Figure 2.6: LIBS measurement and baseline correction for scrap classification [9].

Near-infrared spectroscopy (NIR) uses reflected light to classify materials from absorption patterns linked to molecular bonds. As shown in Figure 2.7, hyperspectral image acquisition combines lighting, imaging, data compression, classification, and an ejection unit for physical separation. The method is non-contact and fast, making it useful for plastics, wood, organic contaminants, coatings, and other polymer-based fractions in industrial recycling and sorting applications. However, NIR mainly measures molecular vibrations rather than elemental composition, so it cannot reliably identify specific metals or alloy grades in recipe-based scrap sorting [20].

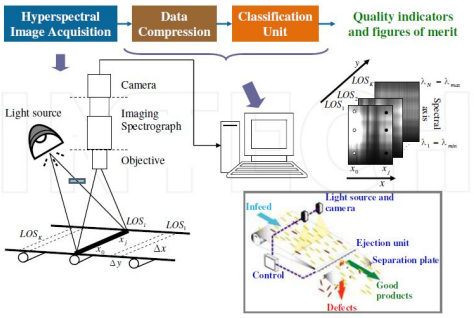


Figure 2.7: NIR sorting system with image acquisition, classification, and ejection unit [20].

Optical or vision-based sorting uses cameras to classify scrap from visible surface features such as shape, size, colour, thickness, and surface condition. As shown in Figure 2.8, cameras capture images of the scrap load, after which deep-learning models detect and label individual pieces or defect categories for grading. This makes the technique attractive for fast, non-contact inspection of categories such as painted, galvanised, oily, oversized, or inclusion-containing scrap. However, because the classification is based on appearance rather than elemental composition, optical sorting cannot directly determine alloy grade or chemical composition. Its performance can also be reduced by occlusion, overlapping pieces, dust, variable illumination, and surface contamination. Optical sensing is therefore best used as a complementary method alongside composition-based sensors in recipe-based scrap sorting [21]. It can also improve process efficiency and reduce manual inspection.

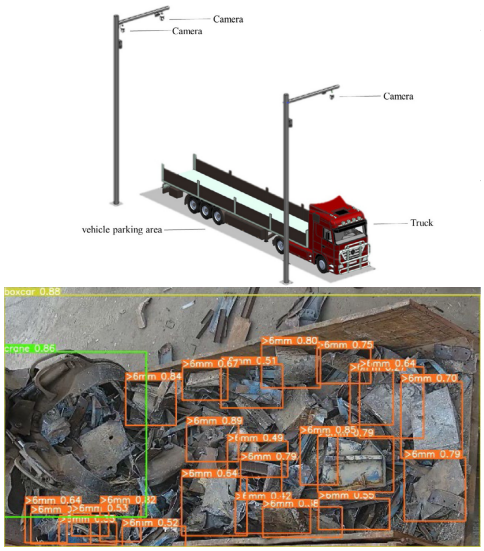


Figure 2.8: Camera-based scrap inspection and deep-learning detection of scrap pieces [21].

The reviewed techniques are summarised in Table 2.1, showing that no single method combines high throughput, low cost, and detailed alloy information. Practical sorting lines therefore often rely on combining complementary sensors. The next section examines how such techniques are integrated into commercial sensor-based sorting systems.

Technique	Detection Principle	Applications	Advantages	Disadvantages
Magnetic [14]	Magnetic attraction	Ferrous / non-ferrous	Very cheap, high throughput, robust	Only ferrous detection
Eddy Current [15]	Induced electric currents in conductive metals	Non-ferrous / non-conductors	Fast, suitable for bulk flows	Performs poorly with small/thin pieces
Induction [18]	Change in electromagnetic field	Detection of metal / non-metal	Cheap, robust, simple	No alloy or elemental information
XRT [19]	Differences in X-ray absorption (density)	Light / heavy metals	Penetrates coatings, distinguishes densities	Expensive, radiation safety required
XRF [8]	Characteristic fluorescence per element under X-ray	Alloy analysis	Very accurate, element-specific	Lower speed, high costs
LIBS [9]	Laser plasma emission spectrum	High-grade alloy separation	High precision, detects light elements	Relatively slow, high energy consumption
NIR [20]	Reflection spectrum	Coatings, plastics, paint on metal	Fast, cheap	Limited applicability to metals themselves
Optical [21]	Shape, color, texture, gloss	Coating, shape recognition	Very fast, can be combined with other techniques	Not chemically specific, sensitive to dirt and dust

Table 2.1: Overview of detection techniques used in sensor-based sorting.

2.3. Sensor-based sorting systems

Modern sensor-based scrap sorting systems are increasingly designed to combine material identification accuracy with industrial throughput. In commercial metal recycling, XRT, XRF, LIBS, optical sensors, and 3D object recognition are commonly combined to separate metals by density, elemental composition, alloy class, or visible surface characteristics. For recipe-based steel production, the most relevant requirement is not only the recognition of broad scrap categories, but the ability to obtain sufficiently reliable compositional information at piece level while maintaining industrial processing capacity.

TOMRA's AUTOSORT™ PULSE illustrates the use of dynamic LIBS for industrial aluminium alloy sorting. The system shown in Figure 2.9, combines LIBS with 3D object scanning, shape recognition, and AI-based object singulation to handle overlapping pieces. TOMRA reports purity levels of 95% or higher for 5xxx and 6xxx aluminium streams, with up to 15 t/h on a 1.2 m belt. Its relevance for this thesis is the demonstrated high-throughput per-piece spectral sorting capability in demanding industrial recycling environments today [22].

Figure 2.9: TOMRA AUTOSORT™ PULSE dynamic LIBS sorting system [22].

Ocean Optics' SpeedSorter, integrated in Austin AI sorting systems, demonstrates LIBS-based in-line sorting of non-ferrous scrap. The system shown in Figure 2.10, measures the chemical composition of individual aluminium pieces and sends the result to the separator. Reported separations include wrought from cast aluminium, aluminium from magnesium, and 6xxx alloys from mixed scrap. The case study reports operation at 3-4 t/h, with cost estimates based on 4-5 t/h [23].

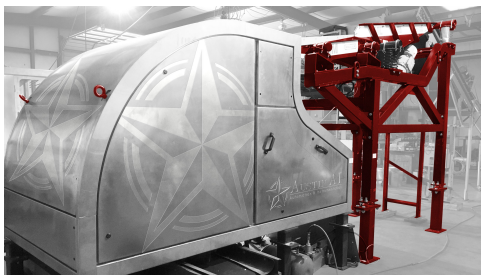


Figure 2.10: Ocean Optics SpeedSorter [23].

STEINERT's PLASMAX LIBS sorter demonstrates how multiple sensing and separation steps can be integrated in one industrial aluminium alloy sorter. As shown in Figure 2.11, the system feeds material onto the belt, performs 3D and in-flight detection, applies multi-spot LIBS measurements, and uses AI-supported evaluation to trigger compressed-air separation. It can discharge three products in one run and is intended for alloy classes such as 3xxx, 4xxx, 5xxx, and 6xxx. STEINERT reports purities above 95% and a processing capacity of up to 11 t/h. Although developed for aluminium rather than steel, it illustrates the industrial feasibility of high-throughput per-piece LIBS classification for recipe-based sorting concepts [24].

Figure 2.11: STEINERT PLASMAX sorting system [24].

SGM Magnetics illustrates a cascaded approach to sensor-based sorting. Its XRT systems separate by density, while XRF-T combines X-ray transmission with fluorescence to add element-based information. For chemical composition analysis, SGM presents LIBS as a complementary but more capital-intensive technology. As shown in Figure 2.12, the Cleansort LIBS concept uses square laser pre-ablation to remove surface contamination before spectral measurement. The modular system is reported for aluminium scrap throughputs between 5.5 and 11 t/h, and can be recalibrated for other metals, including steel [25, 26].

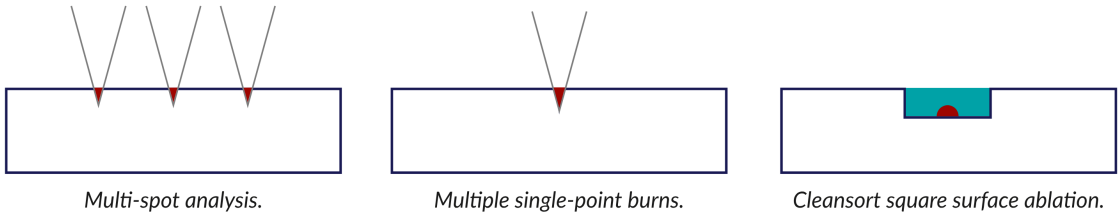


Figure 2.12: SGM Cleansort LIBS square surface ablation before spectral analysis [26].

The commercial systems summarised in Table 2.2 show that high-throughput sensor-based alloy sorting is technically feasible, especially for aluminium and non-ferrous scrap streams. They also show a clear trend towards combining spectral analysis, 3D object detection, AI-based classification, and controlled ejection. However, most reported performance values are supplier-provided and application-dependent. For recipe-based steel scrap sorting, the remaining challenge is not only detecting or classifying individual items, but also using the resulting information in a decision-making strategy. The next section therefore reviews existing academic approaches to automated scrap sorting and optimisation.

System	Accuracy (Purity)	Precision (Analysis)	Throughput (kg/h)
[22] TOMRA AUTOSORT™ PULSE (LIBS)	≥95% sorted alloy purity	Multi-point LIBS scanning (up to 97% purity)	Up to 15 000
[25, 26] SGM Cleansort (LIBS)	>98% purity with XRF/XRT pre-sort	Quantitative per-piece composition analysis	5 500 - 11 000
[24] Steinert PLASMAX (LIBS)	>95% purity in alloy output	“Maximum precision” multi-spot LIBS sorting	Up to 11 000
[23] Ocean Optics/Austin AI SpeedSorter (LIBS)	High-purity alloy sorting (<0.1% Cu, Zn)	Accurate LIBS elemental ID of alloys	3 000 - 4 000

Table 2.2: Overview of elemental analysis sorting machines.

2.4. Existing approaches to automated scrap sorting

Existing approaches to automated scrap sorting cover optimisation, process simulation, sensor-based classification, and machine-vision-based quality inspection. Together, these studies provide important building blocks for automated sorting systems and demonstrate the potential of automation across different stages of the recycling process. However, they generally focus on scrap selection, physical flow, or material classification separately, and do not integrate piece-level compositional information with recipe-based heap allocation within a unified decision-support framework for operational sorting.

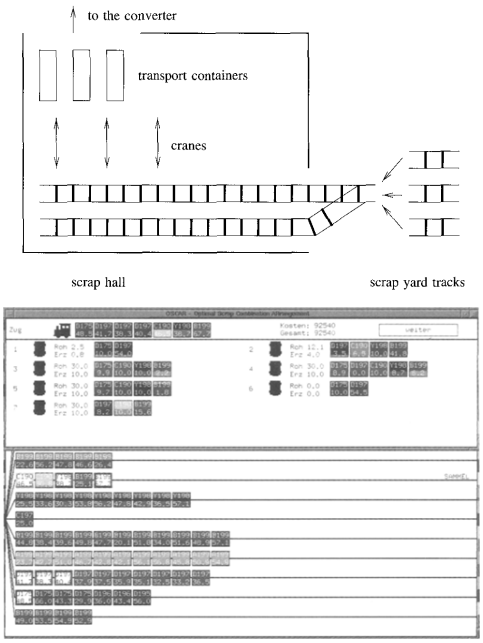


Figure 2.13: Scrap-yard logistics and OSCAR interface for cost-minimising scrap selection [27].

Bernatzki et al. (1998) [27] studied scrap selection for steel production as a mixed-integer programming problem. As shown in Figure 2.13, their approach connects the physical logistics of scrap supply, including scrap-yard tracks, rail cars, cranes, and transport containers, with an optimisation interface that proposes scrap combinations for upcoming converter processes and supports operational planning decisions within the steelmaking production chain. The model minimises material cost while satisfying cooling requirements, iron-content limits, tramp-element constraints, and transport restrictions based on the number and position of selected rail cars. This is relevant for this thesis because it frames scrap use as a constrained optimisation problem rather than a purely operational decision. However, the optimisation is performed at the level of predefined scrap classes and rail-car inventories, without considering detailed variability between individual scrap items. It therefore does not address sensor-characterised individual scrap pieces, dynamic heap formation, or allocation to recipes based on piece-level composition data.

Compañero et al. (2023) [5] evaluated how compositional information affects scrap-based steel production using the expected value of perfect information framework. Their model compares partial information, where scrap composition is only known within specification ranges, with full information, where the exact composition is available before melting. The results show that full information can reduce production costs by 8-10% and make excess alloying costs almost negligible. This is relevant because it quantifies the economic value of recovering composition data from end-of-life scrap. However, the study evaluates scrap-class information and furnace charging, not piece-level sorting or dynamic heap allocation.

Wu et al. (2024) [28] developed a DEM-based virtual experiment model for an automated aluminium scrap sorting facility. As shown in Figure 2.14, the model represents both the process layout, consisting of singulation, sensor scanning, AI classification, and ejection, and the irregular particle geometry obtained from 3D-scanned scrap pieces. The particle shapes were reconstructed using detailed scanning data and subsequently approximated within the DEM environment, allowing realistic representation of scrap behaviour during transport and separation. This makes the study relevant because it shows how particle-scale simulation can support the design of full sorting lines rather than only individual sensors. The results indicate that flow regularisation is essential: high feeding rates increase overlap and clustering, reducing successful scanning and ejection. A feasible feeding range of about 3-4 particles per second achieved a sorting rate of approximately 83%. However, the study does not address recipe-level allocation, mass conversion, or economic value.

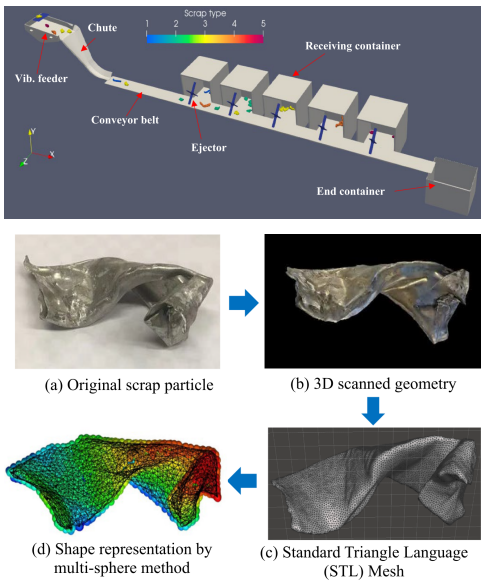


Figure 2.14: DEM-based aluminium scrap sorting line and 3D particle-shape representation [28].

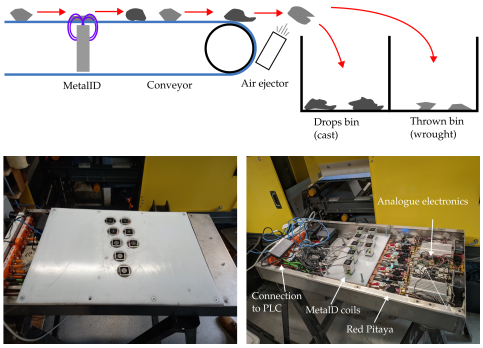


Figure 2.15: Industrial MIS sorting concept and MetalID sensor hardware [29].

Williams et al. (2023) [29] evaluated magnetic induction spectroscopy (MIS) for separating wrought from cast aluminium in shredded Twitch scrap. As shown in Figure 2.15, the industrial setup placed MetalID coils beneath a conveyor and used an air ejector to deflect pieces classified as wrought. The static laboratory system achieved 89.66% recovery at 94.96% purity, but industrial tests at 2 m/s performed worse because pieces only partly covered sensors, orientation changed the response, and algorithm choice affected generalisation. The study highlights the gap between controlled sensor tests and industrial sorting performance.

Díaz-Romero et al. (2023) [30] investigated multi-sensor classification of post-consumer aluminium scrap by fusing LIBS spectra with RGB and depth images using deep learning. As shown in Figure 2.16, the method combines augmented LIBS spectra with visual and 3D representations of the same scrap pieces, allowing chemical, surface, and geometric features to be learned together. The study used 773 aluminium scrap pieces and showed that fusion improved classification: cast/wrought separation reached 99% precision, recall, and F1-score, while three-fraction commercial sorting achieved 86% precision, 83% recall, and 84% F1-score. This is relevant because it shows the value of complementary sensors, although the output remains a class prediction rather than recipe-compliant heap allocation.

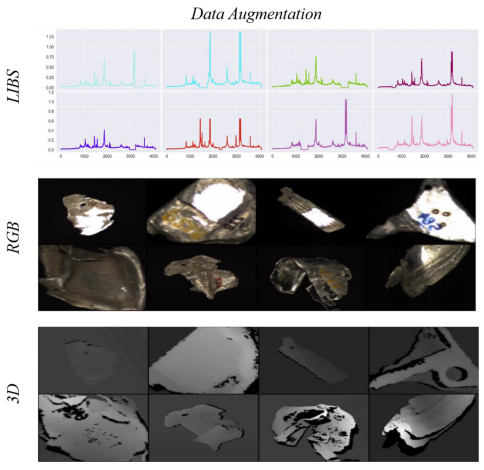


Figure 2.16: Aluminium scrap classification using LIBS, RGB images, and 3D data [30].

Figure 2.17: Steel scrap images before and after CSBFNet visual classification [21].

Xu et al. (2023) [21] developed a deep-learning approach for automatic steel scrap classification and rating using high-resolution camera images. As shown in Figure 2.17, the model detects visible scrap categories in both laboratory and field images, including thickness classes, coatings, oil, inclusions, and overlength pieces. Their CSBFNet architecture combines feature extraction, attention, and multi-scale feature fusion to handle overlapping and irregular scrap. The reported average accuracy is 92.4%, with an mAP of 90.7%. This is directly relevant to ferrous scrap inspection, but the method evaluates visual quality classes rather than chemical composition or recipe compatibility.

The studies reviewed and summarised in Table 2.3 show that automated scrap sorting has advanced through optimisation, process simulation, sensor fusion, and visual quality inspection. However, these approaches mainly optimise scrap classes, classify individual items, or model physical sorting-line behaviour. They do not yet connect piece-level composition data to recipe-based heap formation under mass, capacity, and value constraints. The following section therefore summarises this gap and defines the research direction of this thesis.

Study	Approach	Data and scope	Reported performance
[27] Bernatzki et al. (OR Spektrum, 1998)	Mixed-integer programming (MIP)	Steel production scrap classes with material and transport constraints	Cost-minimising scrap combination; no item or mass conversion metric reported
[5] Compañero et al. (Miner. Econ., 2023)	Expected value of perfect information + raw material optimisation	Scrap-based steel production scenarios with partial vs. full composition information	Production costs reduced by 8-10% with perfect information; excess costs became negligible
[28] Wu et al. (Waste Manag., 2024)	Discrete Element Method (DEM) virtual experiment model	Complex-shaped aluminium scrap particles; automated line with singulation, sensor scanning, and ejection	Feasible feeding range around 3-4 particles/s; sorting rate about 83%
[29] Williams et al. (Sensors, 2023)	Magnetic Induction Spectroscopy (MIS) + machine learning	Shredded mixed aluminium scrap ('Twitch') for cast/wrought separation	Static lab system: 89.66% recovery and 94.96% purity; industrial conveyor performance lower
[30] Díaz-Romero et al. (Resour. Conserv. Recycl., 2023)	LIBS + RGB-D image fusion using deep learning	773 post-consumer aluminium scrap pieces; cast/wrought and three-way commercial classification	Cast/wrought: 99% precision, recall, and F1; three fractions: 86% precision, 83% recall, 84% F1
[21] Xu et al. (Eng. Appl. Artif. Intell., 2023)	CSBFNet deep learning model for machine vision	High-resolution images of steel scrap for multi-category classification and rating	Average accuracy: 92.4%; mAP: 90.7%

Table 2.3: Overview of relevant literature on automated scrap sorting, scrap optimisation, and composition-information value.

2.5. Summary and research gap

The previous sections showed that steel recycling increasingly depends on accurate scrap characterisation and sorting to meet the compositional requirements of modern steel grades. Sensor-based technologies such as LIBS, XRF, XRT, NIR, and optical sorting can support this process by providing information on elemental composition, density-related properties, surface condition, or visual characteristics. Commercial systems from suppliers such as TOMRA, STEINERT, and SGM demonstrate that sensor-based metal sorting can operate at industrially relevant throughputs, with supplier-reported purities above 95% and throughputs in the range of several tonnes per hour (Section 2.3). However,

these systems mainly demonstrate the ability to identify or separate individual pieces into predefined material categories. They do not determine which steel recipe should be targeted, nor how different scrap pieces should be combined to form a chemically feasible and economically valuable heap.

The reviewed academic literature shows a similar limitation. Existing studies have mainly addressed either scrap-mix optimisation, sensor-based classification, or the physical design of automated sorting lines (Section 2.4). Bernatzki et al. [27] formulated scrap selection for steel production as a constrained optimisation problem, but at the level of predefined scrap classes. Compañero et al. [5] demonstrated the economic value of improved compositional information in scrap-based steel production, but did not develop a piece-level allocation method. Wu et al. [28] modelled the physical flow of aluminium scrap in an automated sorting facility using DEM, while Williams et al. [29], Díaz-Romero et al. [30], and Xu et al. [21] focused on sensor-based or vision-based classification of individual scrap items. These studies provide important foundations, but they do not address the subsequent decision problem of how sensor-characterised scrap items should be allocated to heaps that satisfy the chemical constraints of specific steel recipes.

This knowledge gap forms the basis for the Sorting Decision Layer proposed in this thesis. Existing sorting equipment can provide item-level information and perform the physical separation, but an intermediate decision layer is required to translate this information into recipe-based sorting goals. In this thesis, the Sorting Decision Layer is defined as the algorithmic layer between sensor-based item characterisation and physical sorting execution. Its task is to decide which scrap items should be grouped together, which steel recipe each heap should target, and whether the resulting mass-weighted heap composition remains within the lower and upper elemental limits of that recipe.

Such a decision layer requires algorithms because the allocation problem cannot be solved by simple item-by-item classification. A single scrap item may be unsuitable for a recipe on its own, but useful in combination with other items whose compositions compensate for its deviations. Conversely, an item that is chemically acceptable for one recipe may reduce the value or feasibility of another heap. The algorithm therefore has to search through many possible item–heap–recipe combinations while considering chemical feasibility, heap capacity, available sorting space, mass conversion, heap utilisation, and economic value. For large scrap streams and hundreds of possible steel recipes, exhaustive evaluation becomes impractical, so efficient allocation algorithms are required to construct and improve feasible sorting solutions.

This thesis addresses the identified gap by developing and evaluating a recipe-based scrap allocation model. Instead of treating sorting as a classification problem alone, the proposed approach uses piece-level mass and composition data to select combinations of scrap items whose weighted-average elemental composition falls within the lower and upper bounds of target steel recipes. The Sorting Decision Layer therefore connects sensor-based item characterisation with production-oriented objectives, including mass conversion, heap utilisation, and economic value. In this way, the thesis evaluates not only whether scrap items can be identified, but also how they can be allocated to maximise the value of sorted outputs under realistic recipe and capacity constraints.

A further limitation in this research area is the lack of publicly available industrial datasets containing piece-level scrap mass and elemental composition at sufficient scale. Publicly available data generally describe bulk scrap categories, standardised alloy compositions, or finished steel grades, rather than individual scanned scrap pieces suitable for heap formation algorithms. Industrial sensor systems may be capable of generating such data, but raw piece-level sensor outputs are typically not publicly shared. For this reason, this thesis uses synthetically generated scrap data based on steel composition ranges. Although synthetic data cannot fully reproduce all variability of industrial scrap streams, it enables controlled evaluation of recipe-based allocation strategies and provides a reproducible basis for exploring the potential of piece-by-piece scrap sorting before large-scale industrial implementation.

3

Sorting Decision Layer

The literature review (Chapter 2) showed that existing sensor-based sorting systems can increasingly identify the composition or class of individual scrap pieces, but that this information is not yet directly translated into recipe-based steel production decisions. This chapter therefore develops the Sorting Decision Layer as the missing link between piece-level sensor data and operational scrap allocation. Instead of treating sorting as the rejection or acceptance of individual items, the proposed system evaluates how scanned scrap pieces can be combined into heaps whose mass-weighted composition satisfies the chemical limits of target steel recipes. In this way, the chapter turns the research gap identified in the literature into a system architecture, mathematical allocation model, and feasibility-checking procedure.

This chapter explains the intended sorting system by first describing the material flow in Section 3.1. The position and role of the Sorting Decision Layer are then introduced in Section 3.2. Next, Section 3.3 formalises the allocation problem, and Section 3.4 describes the feasibility checks used during heap construction. Together, these sections answer the second sub-research question: "How can a system be developed that allocates scrap metal into sorted scrap streams based on the chemical composition of steel recipes according to European Steel Standards?" in Section 3.5.

3.1. Material flow and sorting process

The intended material flow of the sorting system is displayed in Figure 3.1. Incoming scrap items are spread out on a conveyor and go through the scanning station, where their mass and chemical composition are determined and logged. After the scanning stage, the analysed scrap pieces have to stay in place on the conveyor so they can be identified until enough pieces are scanned to sort them into recipe-compliant heaps. This extended conveyor part, after scanning and before sorting, therefore acts as a buffer.

From the buffer, scrap items can be routed to one of multiple sorting heaps, each corresponding to a target steel recipe. Before scrap items can be added to the heaps, the allocation process evaluates whether a combination of available scrap items can form a valid recipe. A scrap heap is considered feasible if the mass-weighted average composition of all selected scrap pieces satisfies the minimum and maximum limits for every constrained element of the recipe. If the addition of a scrap item would cause any elemental average to violate the recipe bounds, that item is rejected for the current heap.

Scrap allocation is further constrained by a maximum heap mass, representing furnace charging or logistical limits. Once this maximum mass is reached, no further scrap items are added, even if chemically compatible material remains available. Scrap items are treated as indivisible units and can be assigned to at most one heap. After a heap has been formed, the contributing scrap items are removed from the available pool before subsequent heaps are constructed.

The allocation process is performed iteratively. In each iteration, feasible heaps are constructed for all candidate steel grades using the currently available scrap. These candidate heaps are then evaluated based on their associated economic value, defined as the total heap mass multiplied by the grade-specific price. The most valuable heap is selected, finalised, and removed from further consideration. The process is then repeated with the remaining scrap until either all heaps are filled or no additional feasible combinations can be formed. Leftover scrap items are returned through the return line to be reconsidered with future incoming scrap shipments.

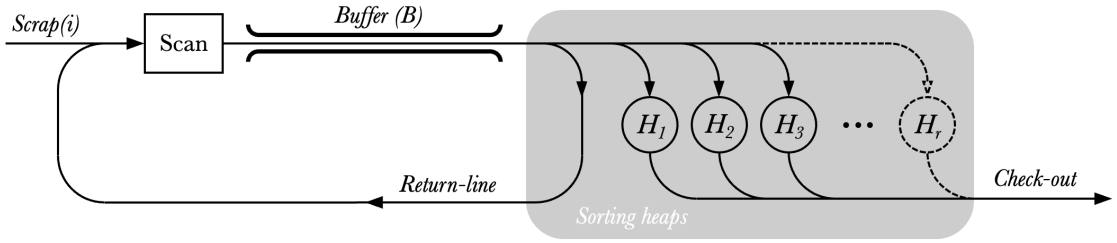


Figure 3.1: Flow diagram of the material flow in the intended scrap sorting system.

3.2. Overview of the Sorting Decision Layer

As discussed in Section 2.3, sensor-based sorting equipment for scrap detection and separation is already commercially available. Therefore, the system developed in this thesis is not intended to replace the physical sorting equipment itself, but to operate as an additional decision-making layer on top of it. This layer is positioned between the equipment layer and the operator layer, where it links measured item data to recipe-based allocation decisions. In this thesis, this developed system is referred to as the ”Sorting Decision Layer”.

Figure 3.2 illustrates the difference between a conventional item-based sorting approach and the recipe-based allocation approach proposed in this thesis. In a conventional approach (A), scrap items are accepted or rejected using strict composition boundaries for the target recipe. As a result, items with individual elemental concentrations outside these bounds are excluded, even if they could still be useful in combination with other scrap pieces. The proposed approach (B) instead evaluates scrap items as part of a combined heap. Individual compositional deviations or impurities can therefore be balanced by other items, as long as the resulting mass-weighted heap composition remains within the lower and upper limits of the selected steel recipe. By using this approach, scrap items that would otherwise

be downgraded or downcycled into less restrictive lower-grade steel can instead be included in an equivalent or even more valuable steel grade. This combination-based decision-making is the central task of the Sorting Decision Layer.

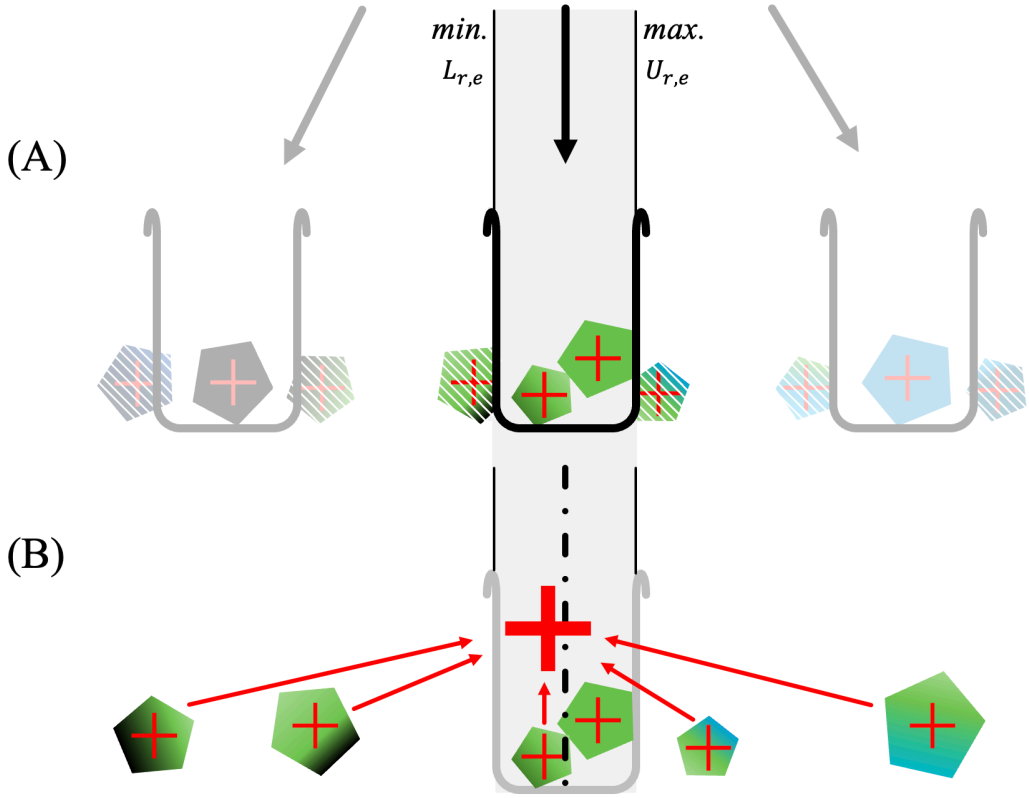


Figure 3.2: (A) Conventional sorting approach with strict composition boundaries, where scrap items are rejected if their individual elemental composition falls outside the limits of the desired recipe. (B) Proposed recipe-based sorting approach, where scrap items are combined so that their compositional deviations or impurities balance each other and the resulting mass-weighted heap composition remains within the limits of the desired recipe.

Figure 3.3 shows the conceptual position of the proposed Sorting Decision Layer within the intended scrap sorting system. The equipment layer detects the elemental composition and mass of individual scrap items and performs the physical separation of the material. The operator layer defines and maintains the available steel recipe library and supervises the overall process. Between these two layers, the Sorting Decision Layer translates the measured item data and recipe information into recipe selections, heap assignments, and sorting commands for the equipment.

Based on the material flow described in Section 3.1, the proposed recipe-based sorting approach explained in Figure 3.2, and the position of the Sorting Decision Layer shown in Figure 3.3, the operation of the proposed sorting system can be summarised as a sequence of allocation steps. The system continuously receives scanned scrap items, temporarily stores them in a buffer, and iteratively allocates them to sorting heaps associated with steel recipes. The overall procedure can be described as follows:

1. **Scrap scanning and data registration.** Incoming scrap items i pass through a scanning station where their mass m_i and elemental composition vector $\{x_{i,e}\}_{e\in\mathcal{E}}$ are measured and recorded. Each scanned item is stored in the buffer as an identifiable unit.
2. **Buffer accumulation.** The scanned scrap items remain on the conveyor buffer until a sufficient number of items are available for allocation. The buffer, therefore, represents the current set of available scrap items $\mathcal{S}$ from which candidate heaps can be formed.
3. **Candidate heap construction.** For each steel recipe $r \in \mathcal{R}$, the system attempts to construct a candidate heap $H_r \subseteq \mathcal{S}$ by incrementally selecting compatible scrap items from the buffer. Items are added one by one as long as the heap satisfies the system constraints.

4. **Feasibility verification.** Each candidate's addition of a scrap item is checked against the system constraints. These include:
   - The maximum allowable heap mass.
   - The elemental composition limits defined by the target recipe.
   - the restriction that each scrap item may be used at most once.

   Only items that satisfy all constraints are accepted into the heap.
5. **Heap valuation.** Once no further scrap items can be added, the resulting heap is evaluated economically. The value of a candidate heap is defined as the product of the total heap mass and the price associated with the corresponding steel recipe.
6. **Selection of the best heap.** All candidate heaps constructed for the available recipes are compared. The heap with the highest economic value is selected and finalised.
7. **Scrap removal and iteration.** The scrap items belonging to the selected heap are removed from the buffer and routed to the corresponding sorting heap. The allocation process is then repeated using the remaining scrap items until no additional feasible heaps can be formed or the available sorting capacity is reached.

The following sections formalise this allocation process. First, Section 3.3 translates the system constraints and objectives into a mathematical framework. Next, Section 3.4 describes the feasibility checks used during heap construction. Finally, Chapter 4 introduces the candidate algorithms that implement the allocation strategy within this framework.

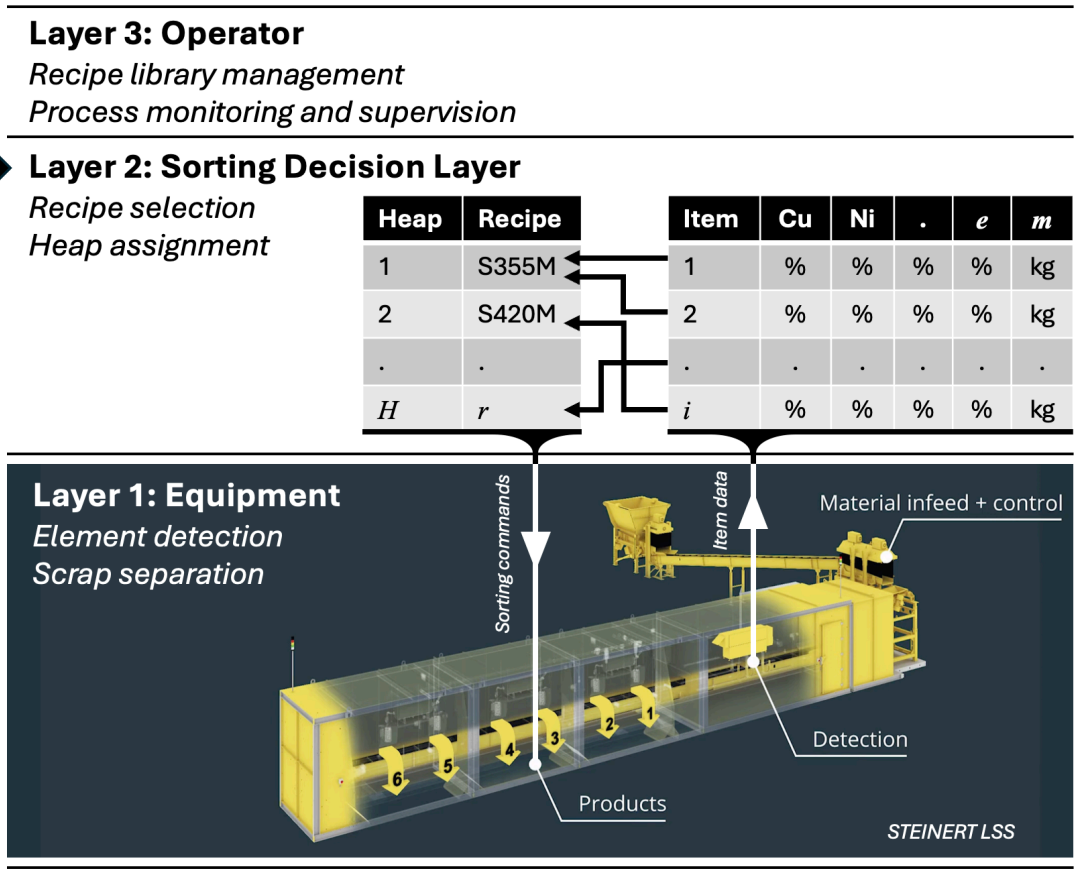


Figure 3.3: Conceptual positioning of the proposed Sorting Decision Layer between the equipment layer and the operator layer. The equipment layer provides measured item data to the decision layer, while the decision layer returns sorting commands for scrap separation. The equipment image is adapted from [31].

3.3. Mathematical model

To simulate the proposed system, the system constraints and targets are translated into a mathematical framework. The structure of this framework is visualised in Figure 3.4, where scrap items are assigned to candidate heaps, candidate heaps are linked to steel recipes, and the main mass, composition, and value constraints are indicated. In this framework, $\mathcal{S}$ represents the set of available scrap items in the buffer. At each allocation cycle, the buffer contains scrap items $i \in \mathcal{S}$, each characterised by a mass m_i and an elemental composition $x_{i,e}$ for every element $e \in \mathcal{E}$.

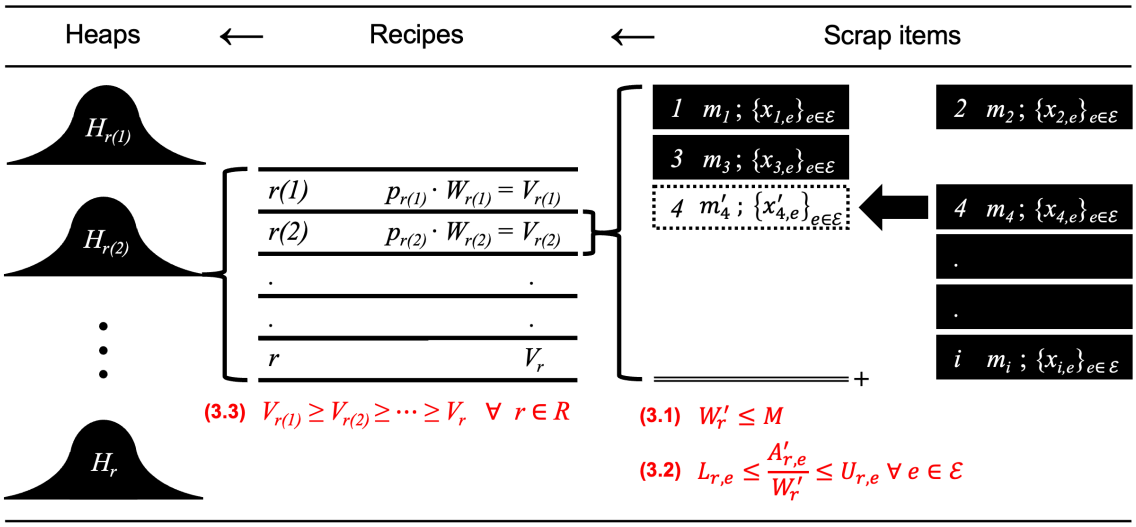


Figure 3.4: Visual representation of the allocation model, showing how scrap items are linked to recipe selection, heap construction, and feasibility constraints, as described in Section 3.2, Section 3.3, and Section 3.4.

For a given steel recipe r, the objective is to construct a heap $H_r \subseteq \mathcal{S}$ that maximises the heap value defined in Equation (3.3), while satisfying both the maximum heap capacity constraint in Equation (3.1) and the compositional constraints in Equation (3.2). These compositional constraints are derived from European steel standards, where each element must lie within a specified range for a valid steel grade.

A heap is feasible if the following conditions hold:

$$\sum_{i \in H_r} m_i \leq M, \tag{3.1}$$

$$L_{r,e} \leq \frac{\sum_{i \in H_r} m_i\, x_{i,e}}{\sum_{i \in H_r} m_i} \leq U_{r,e}, \quad \forall e \in \mathcal{E}, \tag{3.2}$$

where M is the maximum allowable heap mass, and $L_{r,e}$ and $U_{r,e}$ are the lower and upper bounds for element e in recipe r.

Example of constraint evaluation. To illustrate how Equation (3.2) is applied in practice, consider a heap consisting of three scrap items with masses $m_1 = 100$ kg, $m_2 = 200$ kg, and $m_3 = 300$ kg. Suppose the mass fraction of element e (e.g. chromium) in these items is $x_{1,e} = 0.10$, $x_{2,e} = 0.15$, and $x_{3,e} = 0.20$, respectively.

The weighted average composition of element e in the heap is then computed as:

$$\frac{100 \cdot 0.10 + 200 \cdot 0.15 + 300 \cdot 0.20}{100 + 200 + 300} = \frac{10 + 30 + 60}{600} = 0.167.$$

If the recipe requires $L_{r,e} = 0.16$ and $U_{r,e} = 0.18$, the heap satisfies the compositional constraint for this element. If the value falls outside this range, the candidate heap is infeasible, and the item combination must be rejected.

Starting from an empty heap, scrap items are selected incrementally and added to H_r. An item is only added if it does not violate the constraints of Equation (3.1) and Equation (3.2). This process continues until no further items can be added or the mass limit is reached.

The resulting heap value is then computed as:

$$V_r = p_r \sum_{i \in H_r} m_i, \tag{3.3}$$

where p_r is the price associated with recipe r.

For each allocation cycle, this allocation logic is executed for all available recipes. The heap with the highest value is selected, finalised, and its scrap items are removed from the buffer. The process then repeats on the remaining scrap.

3.4. Scrap items selection pseudo-code

Function 1: TryAdd acts as a feasibility gate to consider adding a candidate scrap item i to the current heap H_r of recipe r. The routine verifies whether the item can be added without violating any of the problem constraints. It checks (Check 1) item availability, (Check 2) the maximum heap mass M, and (Check 3) the chemical composition bounds $[L_{r,e}, U_{r,e}]$ for each element $e \in \mathcal{E}$. If any constraint is violated, the routine immediately rejects the candidate and returns **false**. Only when all checks are satisfied are the heap statistics updated and the routine returns **true**. This process is also visualised in Figure 3.4, which shows the heap data framework along with the mathematical constraints discussed in Section 3.3 in red.

The procedure receives the candidate item's mass m_i and elemental composition vector $x_{i,e}$, together with the current heap state. The heap is represented by the selected item set H_r, its total mass

$$W_r = \sum_{j \in H_r} m_j,$$

and the element mass-sums

$$A_{r,e} = \sum_{j \in H_r} m_j x_{j,e}.$$

Tracking the element mass-sums $A_{r,e}$ rather than the concentrations directly allows efficient incremental updates: when an item is tentatively added, only the additional mass contribution $m_i x_{i,e}$ must be added to the corresponding element total.

The first feasibility test prevents duplicate usage of scrap items. The routine checks whether the candidate item i has already been assigned to a heap selected in a previous allocation round t (i.e. $i \in \hat{H}_t$). In addition, the item is rejected if it already appears in the current heap H_r. These checks ensure that each scrap item is used at most once in the final allocation and cannot appear multiple times within the same heap.

The second check verifies the heap capacity constraint. A tentative heap mass

$$W'_r = W_r + m_i$$

is computed. If $W'_r > M$, the candidate item would cause the heap to exceed the maximum allowable mass (heap capacity), and the routine returns **false**.

The third check ensures that the recipe's chemical composition bounds remain satisfied. For each element $e \in \mathcal{E}$, the tentative updated element mass-sum is computed as

$$A'_{r,e} = A_{r,e} + m_i x_{i,e}.$$

The corresponding mass-weighted concentration is then

$$\bar{x}'_{r,e} = \frac{A'_{r,e}}{W'_r}.$$

If the updated concentration violates the lower or upper bound ($\bar{x}'_{r,e} < L_{r,e}$ or $\bar{x}'_{r,e} > U_{r,e}$), the candidate item is rejected immediately.

If the candidate passes all checks, the tentative values are committed. The heap mass is updated to $W_r \leftarrow W'_r$, and the element mass-sums are updated as $A_{r,e} \leftarrow A'_{r,e}$ for all $e \in \mathcal{E}$. The routine then returns **true**, indicating that the item can be safely appended to the heap.

By centralising these feasibility checks in a dedicated routine, the higher-level allocation algorithm can efficiently test candidate additions (Chapter 4) while ensuring that capacity and chemical constraints are always maintained during heap construction.

Function 1: TryAdd: Checks if scrap item fits the item availability, heap capacity, and chemical boundary constraints of the current recipe.

Input: Candidate scrap item i with mass m_i and composition $x_{i,e}$; current heap H_r for recipe r; current total mass $W_r = \sum_{i \in H_r} m_i$; current element mass-sums $A_{r,e} = \sum_{i \in H_r} m_i x_{i,e}$ for all $e \in \mathcal{E}$; max heap mass M; recipe bounds $L_{r,e}, U_{r,e}$ for all relevant $e \in \mathcal{E}$

Output: **true** if scrap item i passes checks states of W_r and $A_{r,e}$ are updated; otherwise **false**

Check 1: Scrap item already in use;
for 1 **to** t **do**
    `// Already assigned to other heap`
    if $i \in \hat{H}_t$ **then**
        return false
if $i \in H_r$ **then**
    `// Already used in current recipe`
    return false

Check 2: Heap capacity violation;
Let $W'_r \leftarrow W_r + m_i$;
if $W'_r > M$ **then**
    return false

Check 3: Mass-weighted composition bounds;
foreach $e \in \mathcal{E}$ **do**
    `// Compute tentative updated element mass-sums`
    Let $A'_{r,e} \leftarrow A_{r,e} + m_i\, x_{i,e}$;
    Let $\bar{x}'_{r,e} \leftarrow \dfrac{A'_{r,e}}{W'_r}$;
    if $\bar{x}'_{r,e} < L_{r,e}$ ***or*** $\bar{x}'_{r,e} > U_{r,e}$ **then**
        return false;

`// Commit: update heap and element mass sums`
Set $W_r \leftarrow W'_r$;
foreach $e \in \mathcal{E}$ **do**
    Set $A_{r,e} \leftarrow A'_{r,e}$;
return true;

3.5. Chapter summary

This chapter developed the proposed Sorting Decision Layer as the missing link between sensor-based scrap characterisation and recipe-based steel production. The material flow in Section 3.1 showed how incoming scrap items are scanned, stored in a temporary buffer, and routed to recipe-specific heaps after allocation. This flow establishes the operational setting of the system: scrap pieces are no longer treated only as broad material classes, but as identifiable items with a measured mass and elemental composition.

Section 3.2 then positioned the Sorting Decision Layer between the physical sorting equipment and the operator layer. The equipment layer provides the measured item data and performs the physical separation, while the operator layer defines the available steel recipes. The Sorting Decision Layer connects these two layers by deciding which scanned scrap items should be assigned to which recipe-specific heap. The proposed approach differs from conventional item-based sorting because individual scrap pieces do not need to satisfy all recipe limits independently. Instead, scrap items are evaluated as part of a combined heap, where compositional deviations can be balanced by other items as long as the final mass-weighted heap composition remains within the recipe boundaries.

The allocation problem was formalised in Section 3.3. A feasible heap was defined as a subset of available scrap items that satisfies both the maximum heap mass constraint and the lower and upper elemental limits of a selected steel recipe. The model also introduced the value objective, where the economic value of a heap is calculated from its total mass and the price of the corresponding recipe. This formalisation translates the practical sorting problem into a constrained allocation problem in which chemical feasibility, heap capacity, and economic value must be evaluated simultaneously.

The feasibility logic required for heap construction was then described in Section 3.4. The TryAdd function checks whether a candidate scrap item can be added to a heap without violating item availability, heap capacity, or recipe composition constraints. By using mass sums and element mass-sums, the function enables efficient incremental updates during heap construction. This makes the feasibility check suitable for repeated use inside higher-level allocation algorithms.

Together, these sections answer the second sub-research question: “How can a system be developed that allocates scrap metal into sorted scrap streams based on the chemical composition of steel recipes according to European Steel Standards?” Such a system can be developed by combining four components: a material-flow structure that scans and buffers individual scrap items, a recipe library based on European steel composition limits, a Sorting Decision Layer that assigns scanned items to recipe-specific heaps, and a mathematical feasibility model that verifies whether each heap satisfies mass and chemical composition constraints. The resulting system converts piece-level mass and composition data into sorted scrap streams whose weighted-average compositions comply with selected steel recipes.

However, this chapter only defined the structure and constraints of the allocation problem. It does not yet determine which search strategy should be used to find good heap combinations efficiently. Because the number of possible scrap-item combinations grows rapidly with buffer size, heap capacity, and number of available recipes, exhaustive evaluation is not practical for large scrap streams. The next Chapter 4 therefore investigates candidate algorithms that can implement the Sorting Decision Layer by constructing feasible heaps, selecting valuable recipe combinations, and improving allocation quality within acceptable computation times.

4

Candidate algorithms

The previous chapter (Chapter 3) defined the Sorting Decision Layer as the system component that converts piece-level scrap data into recipe-based heap allocation decisions. However, defining the allocation constraints is not sufficient by itself, because the number of possible combinations grows rapidly with the number of scrap items, heaps, and steel recipes. The next step in the thesis is therefore to determine which algorithmic approaches can search this solution space efficiently while still producing chemically feasible and economically valuable heaps. This chapter develops that step by comparing exact optimisation concepts with practical heuristic approaches, and by selecting candidate algorithms that can implement the Sorting Decision Layer within acceptable computation times.

This chapter explains the candidate algorithms by first analysing the size of the allocation solution space in Section 4.1. The hardware and simulation software used for implementation are described in Section 4.2. Next, suitable algorithmic approaches are selected in Section 4.3, after which the heap construction and allocation heuristics are introduced in Section 4.4. Together, these sections answer the third sub-research question: "What kind of algorithms are suitable for assigning scrap items within the proposed sorting system, considering solution quality and computational efficiency?" in Section 4.5.

4.1. Problem solution space

The solution space of the proposed scrap sorting system is extremely large due to the combinatorial nature of assigning scrap items to heaps and evaluating all possible recipe-based combinations. Each heap must be filled with a combination of scrap items that satisfies the chemical bounds of one of the 315 predefined steel recipes. There are S available scrap items, each with a unique mass and composition, and H heaps to be filled.

For each heap, the system considers all $R = 315$ recipes. For each recipe, it evaluates all possible subsets of the available scrap items to check whether they meet the recipe's chemical constraints. Since each scrap item may either be included or excluded in a candidate subset, the number of possible subsets per recipe is 2^S, where S is the total number of scrap items. This means that the total solution space $N_{\text{solutions}}$ for every heaps is:

$$N_{\text{solutions}} = R \times 2^S \tag{4.1}$$

Even for modest values such as $S = 400$ scrap items, the total number of possible combinations exceeds 8×10^{120} for a single heap. To put this into perspective: the estimated number of atoms in the observable universe is around 10^{80}, and the number of legal board positions in a chess game is approximately 10^{47}. The solution space faced by the proposed system is therefore not just large, it is astronomically large. Exhaustive enumeration of all combinations is fundamentally infeasible. This highlights the importance of using intelligent search strategies that can efficiently navigate this combinatorial landscape, as will be discussed in Section 4.3.

4.2. Hardware and simulation software

The simulation model used in this study was fully developed in VBA (Visual Basic for Applications) within Microsoft Excel (Microsoft 365 Apps for enterprise, version 2408, Build 17928.20156). This software environment was chosen to allow full transparency and control over each step of the simulation process. By writing each part of the logic in a separate macro, it became easier to check, test, and verify every stage individually. Each intermediate result is written to its own worksheet tab, which makes it possible to inspect the internal states and identify errors or unexpected behaviour early in the development process. Example screenshots of the Excel tabs are provided in Appendix B to illustrate the table structure used throughout the data-generation and allocation procedure.

The simulation was executed on a desktop computer running Windows 11 (version 26200), with an Intel i7-6700K processor, and 16 GB of 3200 MHz DDR4 memory. The hardware provided sufficient speed and stability to run large simulations involving up to 2000 scrap items with multiple heaps. All results presented in Chapter 6 were generated using this setup.

4.3. Suitable algorithms

To answer the third sub-research question, "What kind of algorithms are suitable for assigning scrap items within the proposed sorting system, considering solution quality and computational efficiency?", several algorithmic approaches were considered. The proposed allocation problem has a combinatorial structure similar to multidimensional knapsack problems [32, 33] and scrap-mix optimisation problems [27]. Scrap items are selected under mass-capacity and chemical-composition constraints, while the objective is to maximise the value or utilisation of the resulting heap.

At the methodological selection stage, exact optimisation techniques such as Integer Linear Programming (ILP) [34], Mixed-Integer Linear Programming (MILP) [34], branch-and-bound [35], and branch-and-cut [36] were considered. These methods are attractive because they can provide a proof of optimality when a problem instance is solved to completion. They have also been applied successfully by Bernatzki et al. [27] to related steel-production problems, such as cost-minimising scrap combination for steel heats. However, they are less suitable as the primary solution method for the proposed system. As shown in Section 4.1, the number of possible scrap combinations grows exponentially with the number of available scrap items. In addition, the proposed system must evaluate multiple recipes, repeated heap-construction cycles, and several element-specific constraints. This makes exact optimisation computationally impractical for large-scale simulation runs.

For this reason, heuristic methods were selected as a more practical alternative [32, 33]. Greedy algorithms are particularly suitable for navigating very large combinatorial spaces because they construct solutions through fast, local decisions rather than exhaustive enumeration. In this thesis, greedy selection is used to add feasible scrap items to a candidate heap based on sorting criteria such as mass, value contribution, or recipe feasibility. Such heuristics do not guarantee global optimality, but they can produce feasible solutions within acceptable computation times for the available hardware (Section 4.2).

Within the greedy algorithm family, several variants were explored. The Single-run Greedy approach assigns items through a single pass over the available scrap item list. The Multi-pass Greedy variant repeatedly reprocesses the remaining scrap items until no further feasible additions can be made, which can improve heap utilisation and mass conversion. In addition, local search techniques [37] were tested to refine heap compositions after the initial greedy assignment. These local search steps attempt to improve an existing solution by exchanging assigned scrap items with unassigned alternatives, thereby exploring valuable combinations that may be missed by a purely greedy construction heuristic. Local search is widely used for difficult combinatorial optimisation problems because it can improve solution quality without requiring complete enumeration of the solution space.

Table 4.1 provides a qualitative comparison of the considered algorithm types in terms of solution quality, computational efficiency, and scalability.

Algorithm	Solution Quality	Efficiency	Scalability
Integer Linear Programming (ILP)	Optimal if solved	Low	Very low
Mixed-Integer Linear Programming (MILP)	Optimal if solved	Low	Very low
Branch-and-bound / branch-and-cut	Optimal if solved	Very low	Very low
Single-run Greedy	Medium	High	High
Multi-pass Greedy	Medium–High	High	High
Greedy + Local Search	High	Medium–High	High

Table 4.1: Qualitative comparison of algorithmic approaches for scrap heap allocation.

In summary, exact algorithms are not feasible as the main solution method for the large and repeatedly evaluated allocation problem considered in this thesis. Greedy heuristics, enhanced with multi-pass construction and local optimisation, provide a scalable compromise between solution quality and computational efficiency. The implementation and inner workings of these heuristics are discussed in Section 4.4.

4.4. Heap construction and allocation heuristics

This section presents the heuristic algorithms used to construct scrap heaps and iteratively allocate scrap items to steel recipes. The allocation problem involves multiple constraints, including heap capacity, chemical composition limits, and the restriction that each scrap item can only be used once. Due to the combinatorial nature of the problem, exact optimisation approaches become computationally expensive for realistic problem sizes. Therefore, several heuristic methods are proposed that aim to efficiently construct high-quality feasible solutions.

All heap construction procedures rely on a common feasibility routine, Function 1: TryAdd, which verifies whether a candidate scrap item can be added to a heap without violating any constraints. This routine ensures compliance with heap capacity limits, chemical composition bounds, and item availability across allocation rounds.

Based on this feasibility check, several greedy heap construction heuristics are developed. The simplest approach constructs heaps using a single greedy pass over the scrap items, while more advanced variants perform multiple passes or use randomised starting orders to explore alternative packing configurations. These approaches aim to improve the utilisation of heap capacity and increase the economic value of the resulting scrap mixtures.

To further improve the solutions produced by the greedy procedures, a local search improvement method is introduced. This method attempts to replace individual items in a heap and greedily repack

the remaining capacity to increase the total heap mass while maintaining feasibility.

Finally, hybrid heuristics are constructed by combining greedy heap construction with the local search improvement step. These hybrid methods aim to benefit from the computational efficiency of greedy construction while allowing additional refinement of the resulting heaps. The following subsections describe the greedy construction heuristics, local search improvement procedure, and the combined hybrid algorithms in detail.

4.4.1. Single-run Greedy

Algorithm 2: Single-run Greedy describes the main heuristic used to construct candidate scrap heaps and iteratively allocate scrap items to recipes. The algorithm follows a single-run greedy strategy in which heaps are constructed for all recipes, evaluated according to their economic value, and the best-performing heap is selected in each allocation round.

The algorithm takes as input the set of steel recipes $\mathcal{R}$, each associated with a price p_r and chemical composition bounds $[L_{r,e}, U_{r,e}]$ for each element $e \in \mathcal{E}$. The available scrap items form the set $\mathcal{S}$, where each item i has a mass m_i and elemental composition $x_{i,e}$. In addition, a maximum heap mass M and the number of allocation rounds T are specified. The algorithm produces two types of result tables: candidate heap tables $\mathcal{T}_{4,t}$ for each round and a final annotation table $\mathcal{T}_5$ containing the selected allocations. Screenshots illustrating the Excel implementation of these tables are shown in Figures B.4 and B.5.

To improve packing efficiency, the scrap items are first sorted in descending order of their mass m_i. This ordering encourages the algorithm to place large scrap items early in the heap formation process, reducing the risk that large pieces remain unused due to capacity limitations.

The algorithm proceeds over T allocation rounds, one for every heap. At the start of each round t, a new candidate heap table $\mathcal{T}_{4,t}$ is initialised. For every recipe $r \in \mathcal{R}$, a candidate heap is constructed using a greedy procedure. The heap is initialised with an empty item set H_r, total mass $W_r = 0$, and element mass-sums $A_{r,e} = 0$ for all $e \in \mathcal{E}$.

Next, the algorithm iterates over all scrap items $i \in \mathcal{S}$ in the predefined sorted order. For each item, the feasibility routine Function 1: TryAdd is called to verify whether the item can be added to the current heap without violating any constraints. This routine checks item availability, heap capacity, and the recipe's chemical composition limits. If the routine returns **true**, the item is appended to the heap H_r, and the corresponding heap mass W_r and element mass-sums $A_{r,e}$ are updated.

After all scrap items have been considered, the total value of the constructed heap is calculated as

$$V_r = p_r \cdot W_r,$$

which represents the economic value of producing steel recipe r from the assembled scrap heap. The tuple (r, H_r, W_r, V_r) is then stored in the candidate result table $\mathcal{T}_{4,t}$.

Once candidate heaps have been generated for all recipes, the rows of $\mathcal{T}_{4,t}$ are sorted in descending order of their value V_r. The top-ranked row corresponds to the most valuable feasible heap found in the current round. Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ denote this selected heap.

The scrap items contained in the selected heap $\hat{H}_t$ are then permanently assigned to recipe $\hat{r}_t$. Each item $i \in \hat{H}_t$ is annotated in the allocation table $\mathcal{T}_5$ with the selected recipe $\hat{r}_t$ and its associated value $\hat{V}_t$. These annotations ensure that the same scrap item cannot be used again in subsequent rounds.

By repeating this process for T rounds, the algorithm incrementally assigns scrap items to the most valuable feasible heaps. The greedy nature of the method allows candidate heaps for all recipes to be constructed efficiently while ensuring that heap capacity and chemical composition constraints are respected at every step.

Algorithm 1: Heap Formation and Iterative Allocation (Single-run Greedy)

Input: Recipes $\mathcal{R}$ with price p_r and chemical bounds $L_{r,e}, U_{r,e}$; Scrap items $\mathcal{S}$ with masses m_i and compositions $x_{i,e}$; Maximum heap mass M; Number of heaps T

Output: Result tables $\mathcal{T}_{4,t}$ with scrap item allocation possibilities H_r are ranked according to their value V_r; The scrap items in the recipe with the highest value $i \in \hat{H}_t$ are annotated in $\mathcal{T}_5$ with their recipe $\hat{r}_t$ and value $\hat{V}_t$.

Sort scrap items $i \in \mathcal{S}$ by descending mass m_i;
Initialize empty result table $\mathcal{T}_5$;
for $t \leftarrow 1$ **to** T **do**
    Initialize empty result table $\mathcal{T}_{4,t}$;
    foreach *recipe* $r \in \mathcal{R}$ **do**
        Initialize heap $H_r \leftarrow \emptyset$, total mass $W_r \leftarrow 0$;
        Initialize element mass sums $A_{r,e} \leftarrow 0 \; \forall e \in \mathcal{E}$;
        foreach *scrap item* $i \in \mathcal{S}$ *in sorted order* **do**
            if TryAdd$(i, H_r, W_r, \{A_{r,e}\}_{e\in\mathcal{E}}, r, M) =$ ***true*** **then**
                Add i to H_r; update W_r and $\{A_{r,e}\}_{e\in\mathcal{E}}$;
        Compute $V_r \leftarrow p_r \cdot W_r$;
        Append (r, H_r, W_r, V_r) in result table $\mathcal{T}_{4,t}$;
    Sort $\mathcal{T}_{4,t}$ in descending order of V_r;
    Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ be the top-ranked row of $\mathcal{T}_{4,t}$;
    foreach *scrap item* $i \in \hat{H}_t$ **do**
        Annotate item i with recipe $\hat{r}_t$ and value $\hat{V}_t$ in $\mathcal{T}_5$;

4.4.2. Multi-pass Greedy

Algorithm 3: Multi-pass Greedy presents an extension of the greedy heap formation procedure in which multiple passes over the scrap items are performed when constructing candidate heaps. The main difference with the single-pass greedy approach is that the algorithm repeatedly scans the scrap item list until no additional item can be added to the heap. This multi-pass strategy allows the heuristic to incorporate smaller or composition-balancing scrap items that may only become feasible after earlier items have been added.

The algorithm takes as input the set of steel recipes $\mathcal{R}$ with their associated economic values p_r and chemical composition bounds $[L_{r,e}, U_{r,e}]$ for each element $e \in \mathcal{E}$. The available scrap items are represented by the set $\mathcal{S}$, where each item i has a mass m_i and elemental composition $x_{i,e}$. In addition, a maximum heap mass M and the number of allocation rounds T are specified. As in the single-run greedy algorithm, the procedure produces candidate heap tables $\mathcal{T}_{4,t}$ for each round and a final annotation table $\mathcal{T}_5$ containing the selected allocations. Screenshots illustrating the Excel implementation of these tables are shown in Figures B.4 and B.5.

At the start of the algorithm, all scrap items $i \in \mathcal{S}$ are sorted in descending order of their mass m_i. This ordering encourages the placement of larger items early in the heap construction process, improving the utilisation of the available heap capacity. An empty allocation table $\mathcal{T}_5$ is then initialised to record the final assignment of scrap items to recipes.

The algorithm proceeds iteratively for T allocation rounds, one for every heap. In each round t, a new candidate heap table $\mathcal{T}_{4,t}$ is initialised. For every recipe $r \in \mathcal{R}$, a candidate heap is constructed using a greedy packing procedure. The heap is initialised with an empty set of items H_r, total mass $W_r = 0$,

and element mass-sums $A_{r,e} = 0$ for all $e \in \mathcal{E}$.

The heap construction then enters a repeated scanning phase. During each pass, the algorithm iterates over all scrap items $i \in \mathcal{S}$ in the predefined sorted order. For each item, the feasibility routine Function 1: TryAdd is invoked to determine whether the item can be added to the current heap without violating any constraints. This routine verifies item availability, heap capacity, and compliance with the chemical composition limits of recipe r. If the routine returns **true**, the item is added to the heap H_r, and the heap mass W_r and element mass-sums $A_{r,e}$ are updated accordingly.

After completing a full pass over all scrap items, the algorithm checks whether at least one item was added during that pass. If new items were successfully inserted, another pass over the scrap items is performed. The process continues until an entire pass is completed without adding any additional items to the heap. At that point, the heap is considered saturated under the current constraints.

Once heap construction is complete for recipe r, the economic value of the heap is computed as

$$V_r = p_r \cdot W_r,$$

representing the value of producing recipe r with the assembled scrap mixture. The tuple (r, H_r, W_r, V_r) is then appended to the candidate result table $\mathcal{T}_{4,t}$.

Algorithm 2: Heap Formation and Iterative Allocation (Multi-pass Greedy)

Input: Recipes $\mathcal{R}$ with price p_r and chemical bounds $L_{r,e}, U_{r,e}$; Scrap items $\mathcal{S}$ with masses m_i and compositions $x_{i,e}$; Maximum heap mass M; Number of heaps T

Output: Result tables $\mathcal{T}_{4,t}$ with scrap item allocation possibilities H_r are ranked according to their value V_r; The scrap items in the recipe with the highest value $i \in \hat{H}_t$ are annotated in $\mathcal{T}_5$ with their recipe $\hat{r}_t$ and value $\hat{V}_t$.

Sort scrap items $i \in \mathcal{S}$ by descending mass m_i;
Initialize empty result table $\mathcal{T}_5$;
for $t \leftarrow 1$ **to** T **do**
    Initialize empty result table $\mathcal{T}_{4,t}$;
    foreach *recipe* $r \in \mathcal{R}$ **do**
        Initialize heap $H_r \leftarrow \emptyset$, total mass $W_r \leftarrow 0$;
        Initialize element mass sums $A_{r,e} \leftarrow 0 \; \forall e \in \mathcal{E}$;
        repeat
            `// Make multiple passes until no additional item is added`
            foreach *scrap item* $i \in \mathcal{S}$ *in sorted order* **do**
                if TryAdd$(i, H_r, W_r, \{A_{r,e}\}_{e\in\mathcal{E}}, r, M) =$ ***true*** **then**
                    Add i to H_r; update W_r and $\{A_{r,e}\}_{e\in\mathcal{E}}$;
        until ***no item was added to*** H_r;
        Compute $V_r \leftarrow p_r \cdot W_r$;
        Append (r, H_r, W_r, V_r) in result table $\mathcal{T}_{4,t}$;
    Sort $\mathcal{T}_{4,t}$ in descending order of V_r;
    Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ be the top-ranked row of $\mathcal{T}_{4,t}$;
    foreach *scrap item* $i \in \hat{H}_t$ **do**
        Annotate item i with recipe $\hat{r}_t$ and value $\hat{V}_t$ in $\mathcal{T}_5$;

After candidate heaps have been generated for all recipes, the rows of $\mathcal{T}_{4,t}$ are sorted in descending order of their value V_r. The highest-ranked row corresponds to the most valuable feasible heap found in the current round. Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ denote this selected heap.

The scrap items contained in the selected heap $\hat{H}_t$ are then permanently assigned to recipe $\hat{r}_t$. Each item $i \in \hat{H}_t$ is annotated in the allocation table $\mathcal{T}_5$ with the selected recipe $\hat{r}_t$ and the corresponding value $\hat{V}_t$. These annotations ensure that the same scrap items cannot be reused in later allocation rounds.

By repeating this process for T rounds, the algorithm gradually assigns scrap items to the most valuable feasible heaps. Compared to the single-pass greedy approach, the multi-pass strategy increases the likelihood of filling remaining capacity with compatible scrap items, potentially improving heap utilisation and overall economic value.

4.4.3. Multi-start Greedy

Algorithm 4: Multi-start Greedy introduces a multi-start greedy heuristic for constructing candidate scrap heaps. In contrast to the deterministic greedy approaches, this method performs multiple randomised heap construction attempts for each recipe and retains the best-performing heap. The goal of the multi-start strategy is to reduce the sensitivity of greedy packing to the ordering of scrap items and thereby improve the likelihood of finding a higher-quality heap configuration.

The algorithm takes as input the set of steel recipes $\mathcal{R}$ with their associated economic values p_r and chemical composition bounds $[L_{r,e}, U_{r,e}]$ for each element $e \in \mathcal{E}$. The available scrap items are represented by the set $\mathcal{S}$, where each item i has a mass m_i and elemental composition $x_{i,e}$. In addition, the maximum allowable heap mass M, the number of allocation rounds T, and the number of random starting configurations N_{starts} are specified. As in the previous greedy algorithms, candidate heap tables $\mathcal{T}_{4,t}$ are generated for each round, while the final scrap allocations are recorded in the annotation table $\mathcal{T}_5$. Screenshots illustrating the Excel implementation of these tables are shown in Figures B.4 and B.5.

At the start of the algorithm, the allocation table $\mathcal{T}_5$ is initialised. The algorithm then proceeds through T allocation rounds. In each round t, a new candidate heap table $\mathcal{T}_{4,t}$ is created. For every recipe $r \in \mathcal{R}$, the algorithm attempts to construct a feasible scrap heap through a series of randomised greedy searches.

The heap is initially empty, with $H_r = \emptyset$, total mass $W_r = 0$, and element mass-sums $A_{r,e} = 0$ for all $e \in \mathcal{E}$. For each of the N_{starts} starting attempts, a temporary copy of the heap state is created, denoted by $(\tilde{H}, \tilde{W}, \{\tilde{A}_e\})$. The list of scrap items $i \in \mathcal{S}$ is then randomly shuffled to generate a new candidate ordering.

Next, the algorithm performs a greedy construction pass over the shuffled scrap items. For each item i, the feasibility routine Function 1: TryAdd is called to determine whether the item can be added to the temporary heap without violating the heap capacity constraint or the chemical composition bounds of recipe r. If the item satisfies all constraints, it is added to the temporary heap $\tilde{H}$, and the corresponding heap mass $\tilde{W}$ and element mass-sums $\tilde{A}_e$ are updated.

After processing all scrap items in the shuffled order, the resulting temporary heap is evaluated. If the temporary heap mass $\tilde{W}$ exceeds the mass of the currently stored heap W_r, the temporary configuration replaces the current best heap. In this way, the algorithm retains the heap with the largest feasible mass among the randomised construction attempts.

Once all N_{starts} attempts have been completed for recipe r, the value of the resulting heap is calculated as

$$V_r = p_r \cdot W_r,$$

representing the economic value of producing recipe r with the assembled scrap mixture. The tuple (r, H_r, W_r, V_r) is then appended to the candidate result table $\mathcal{T}_{4,t}$.

After candidate heaps have been generated for all recipes, the rows of $\mathcal{T}_{4,t}$ are sorted in descending order of their value V_r. The highest-ranked row corresponds to the most valuable feasible heap identified in the current allocation round. Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ denote this selected heap.

Finally, the scrap items contained in the selected heap $\hat{H}_t$ are permanently assigned to recipe $\hat{r}_t$. Each item $i \in \hat{H}_t$ is annotated in the allocation table $\mathcal{T}_5$ with the selected recipe $\hat{r}_t$ and its corresponding value $\hat{V}_t$. These annotations prevent the reuse of the same scrap items in subsequent allocation rounds.

By performing multiple randomised heap construction attempts, the multi-start greedy heuristic explores a larger portion of the solution space than the deterministic greedy approaches. This increases the probability of discovering heaps with higher mass utilisation and, therefore, higher economic value.

Algorithm 3: Heap Formation and Iterative Allocation (Multi-start Greedy)

Input: Recipes $\mathcal{R}$ with price p_r and chemical bounds $L_{r,e}, U_{r,e}$; Scrap items $\mathcal{S}$ with masses m_i and compositions $x_{i,e}$; Maximum heap mass M; Number of heaps T; Number of random starts N_{starts}

Output: Result tables $\mathcal{T}_{4,t}$ with scrap item allocation possibilities H_r are ranked according to their value V_r; The scrap items in the recipe with the highest value $i \in \hat{H}_t$ are annotated in $\mathcal{T}_5$ with their recipe $\hat{r}_t$ and value $\hat{V}_t$.

Initialize empty result table $\mathcal{T}_5$;
for $t \leftarrow 1$ **to** T **do**
  Initialize empty result table $\mathcal{T}_{4,t}$;
  foreach *recipe* $r \in \mathcal{R}$ **do**
    Initialize heap $H_r \leftarrow \emptyset$, total mass $W_r \leftarrow 0$;
    Initialize element mass sums $A_{r,e} \leftarrow 0 \; \forall e \in \mathcal{E}$;
    foreach $s \leftarrow 1$ **to** N_{starts} **do**
      $(\tilde{H}, \tilde{W}, \{\tilde{A}_e\}) \leftarrow (H_r, W_r, \{A_{r,e}\})$ `// Make a copy`
      Shuffle scrap items $i \in \mathcal{S}$;
      foreach *scrap item* $i \in \mathcal{S}$ *in shuffled order* **do**
        if TryAdd$(i, \tilde{H}, \tilde{W}, \{\tilde{A}_e\}_{e\in\mathcal{E}}, r, M) =$ ***true*** **then**
          Add i to $\tilde{H}$; update $\tilde{W}$ and $\{\tilde{A}_e\}_{e\in\mathcal{E}}$;
      if $\tilde{W} > W$ **then**
        `// Save best shuffle by highest mass`
        $(H, W, \{A_e\}) \leftarrow (\tilde{H}, \tilde{W}, \{\tilde{A}_e\})$;
    Compute $V_r \leftarrow p_r \cdot W_r$;
    Append (r, H_r, W_r, V_r) in result table $\mathcal{T}_{4,t}$;
  Sort $\mathcal{T}_{4,t}$ in descending order of V_r;
  Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ be the top-ranked row of $\mathcal{T}_{4,t}$;
  foreach *scrap item* $i \in \hat{H}_t$ **do**
    Annotate item i with recipe $\hat{r}_t$ and value $\hat{V}_t$ in $\mathcal{T}_5$;

4.4.4. Local Search

Function 2: Local Search presents an improvement procedure that is applied to an existing scrap heap to increase its mass utilisation while maintaining all feasibility constraints. The method attempts to improve the current heap configuration by replacing individual scrap items with alternative items from the available scrap set and subsequently repacking the heap using a greedy strategy.

The procedure takes as input a recipe r, the current heap H_r, its total mass W_r, and the element mass-sums $A_{r,e}$ that describe the current chemical composition of the heap. In addition, the set of scrap

items $\mathcal{S}$, the maximum heap mass M, and a maximum number of search iterations $I_{\max}$ are provided. The algorithm returns an improved heap configuration $(H_r, W_r, \{A_{r,e}\})$ if a better feasible combination of scrap items is found.

The local search proceeds iteratively for at most $I_{\max}$ iterations. At the start of each iteration, a Boolean variable *improved* is initialised to **false**. The algorithm then systematically explores possible single-item replacement moves within the current heap.

Function 2: LocalSearch$(r, H_r, W_r, \{A_{r,e}\}, \mathcal{S}, M, I_{\max})$

Input: Recipe r; current heap H_r with total mass W_r and element mass-sums $A_{r,e}$ for all $e \in \mathcal{E}$; scrap item set $\mathcal{S}$ with masses m_i and compositions $x_{i,e}$; maximum heap mass M; maximum number of iterations $I_{\max}$

Output: Improved heap $(H_r, W_r, \{A_{r,e}\})$ that maximizes heap mass while satisfying capacity and chemical constraints

for $iter \leftarrow 1$ **to** $I_{\max}$ **do**
  $improved \leftarrow$ **false**;
  foreach $j \in H_r$ **do**
    foreach $k \in \mathcal{S}$ **do**
      if $k \in H_r$ **then**
        continue
      $(\tilde{H}_r, \tilde{W}_r, \{\tilde{A}_{r,e}\}) \leftarrow (H_r, W_r, \{A_{r,e}\})$ with j removed;
      if TryAdd$(k, \tilde{H}_r, \tilde{W}_r, \{\tilde{A}_{r,e}\}, r, M) =$ ***false*** **then**
        continue
      repeat
        `// Try add other items to heap`
        foreach $i \in \mathcal{S}$ *in descending mass order* **do**
          if $i \notin \tilde{H}_r$ ***and*** TryAdd$(i, \tilde{H}_r, \tilde{W}_r, \{\tilde{A}_{r,e}\}, r, M)$ **then**
            Add i to $\tilde{H}_r$;
      until ***no item was added***;
      if $\tilde{W}_r > W_r$ **then**
        $(H_r, W_r, \{A_{r,e}\}) \leftarrow (\tilde{H}_r, \tilde{W}_r, \{\tilde{A}_{r,e}\})$;
        $improved \leftarrow$ **true**;
        break
    if $improved$ **then**
      break
  if ***not*** $improved$ **then**
    break
return $(H_r, W_r, \{A_{r,e}\})$;

For every item j contained in the heap H_r, the algorithm considers replacing this item with a candidate item k from the full scrap set $\mathcal{S}$. Items that are already present in the heap are skipped to avoid duplicate usage. A temporary heap configuration $(\tilde{H}, \tilde{W}, \{\tilde{A}_e\})$ is created by copying the current heap state and removing item j from it. This temporary configuration represents the starting point for evaluating a

potential replacement.

Next, the candidate item k is tested using the feasibility routine Function 1: TryAdd. This routine verifies that adding the item does not violate the heap capacity constraint or the chemical composition bounds of recipe r. If the candidate item fails the feasibility check, the algorithm continues with the next candidate.

If the candidate item is feasible, it is added to the temporary heap. The algorithm then attempts to further improve the heap by greedily inserting additional scrap items. This is done by repeatedly scanning the scrap set in descending order of mass and attempting to add items that are not yet present in the temporary heap. Each potential addition is evaluated using the Function 1: TryAdd feasibility routine. This repeated scanning continues until a full pass over the scrap items results in no additional insertions, indicating that the temporary heap is saturated under the current constraints.

After this greedy refill step, the temporary heap mass $\tilde{W}_r$ is compared with the mass of the current heap W_r. If the temporary heap has a larger total mass, the new configuration replaces the current heap, and the variable *improved* is set to **true**. The algorithm then breaks out of the inner loops and begins a new iteration using the improved heap as the new starting point.

If no improving replacement is found during a complete exploration of all candidate swaps, the variable *improved* remains **false**. In that case, the algorithm terminates early because no further local improvements are possible.

The procedure ultimately returns the best heap configuration discovered during the search. By performing local replacement moves followed by greedy repacking, the algorithm can escape suboptimal greedy solutions and potentially achieve higher heap utilisation while still respecting the capacity and chemical composition constraints.

4.4.5. Multi-pass Greedy + Local Search

Algorithm 6: Multi-pass Greedy + Local Search combines the multi-pass greedy heap construction procedure with a local search improvement step. The objective of this hybrid heuristic is to first construct feasible scrap heaps using an efficient greedy packing strategy and then refine these heaps by performing targeted local improvements. By combining these two techniques, the algorithm aims to increase heap utilisation and economic value while maintaining all capacity and chemical composition constraints.

The algorithm takes as input the set of steel recipes $\mathcal{R}$ with their associated prices p_r, and chemical composition bounds $[L_{r,e}, U_{r,e}]$ for each element $e \in \mathcal{E}$. The available scrap items are represented by the set $\mathcal{S}$, where each item i has a mass m_i and elemental composition $x_{i,e}$. In addition, the maximum heap mass M and the number of allocation rounds T are specified. As in the previous algorithms, candidate heap tables $\mathcal{T}_{4,t}$ are generated for each round, while the final assignment of scrap items to recipes is recorded in the annotation table $\mathcal{T}_5$.

At the start of the algorithm, the scrap items $i \in \mathcal{S}$ are sorted in descending order of their mass m_i. This ordering encourages large scrap items to be placed early in the heap construction process, improving the utilisation of the available heap capacity. An empty allocation table $\mathcal{T}_5$ is then initialised.

The algorithm proceeds through T allocation rounds, one for every heap. At the beginning of each round t, a new candidate heap table $\mathcal{T}_{4,t}$ is created. For every recipe $r \in \mathcal{R}$, a candidate scrap heap is constructed using the multi-pass greedy packing procedure.

The heap is initialised with an empty item set H_r, total mass $W_r = 0$, and element mass-sums $A_{r,e} = 0$ for all elements $e \in \mathcal{E}$. The algorithm then repeatedly scans the sorted list of scrap items and attempts to add items to the heap using the feasibility routine Function 1: TryAdd. This routine verifies that adding a candidate item does not violate the heap capacity constraint or the chemical composition bounds of recipe r. Each time a feasible item is found, it is added to the heap, and the corresponding heap statistics are updated. The scanning process continues until a complete pass over the scrap items results in no additional insertions, indicating that the heap cannot be further expanded using the greedy strategy.

After the greedy heap construction phase, the resulting heap configuration is further refined using the local search procedure described in Function 2: Local Search. The local search attempts to improve

the heap by replacing individual scrap items with alternative items and subsequently repacking the heap to increase the total mass while maintaining feasibility. If an improved configuration is found, the heap variables $(H_r, W_r, \{A_{r,e}\})$ are updated accordingly.

Once the final heap configuration for recipe r has been determined, its economic value is calculated as

$$V_r = p_r \cdot W_r,$$

which represents the value of producing recipe r with the assembled scrap mixture. The tuple (r, H_r, W_r, V_r) is then appended to the candidate heap table $\mathcal{T}_{4,t}$.

After candidate heaps have been constructed for all recipes, the rows of $\mathcal{T}_{4,t}$ are sorted in descending order of their value V_r. The highest-ranked row corresponds to the most valuable feasible heap found in the current allocation round. Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ denote this selected heap.

Finally, the scrap items contained in the selected heap $\hat{H}_t$ are permanently assigned to recipe $\hat{r}_t$. Each item $i \in \hat{H}_t$ is annotated in the allocation table $\mathcal{T}_5$ with the selected recipe $\hat{r}_t$ and the corresponding value $\hat{V}_t$. These annotations prevent the same scrap items from being used in subsequent allocation rounds.

Algorithm 4: Heap Formation and Iterative Allocation (Multi-pass Greedy + Local Search)

Input: Recipes $\mathcal{R}$ with price p_r and chemical bounds $L_{r,e}, U_{r,e}$; Scrap items $\mathcal{S}$ with masses m_i and compositions $x_{i,e}$; Maximum heap mass M; Number of heaps T

Output: Result tables $\mathcal{T}_{4,t}$ with scrap item allocation possibilities H_r are ranked according to their value V_r; The scrap items in the recipe with the highest value $i \in \hat{H}_t$ are annotated in $\mathcal{T}_5$ with their recipe $\hat{r}_t$ and value $\hat{V}_t$.

Sort scrap items $i \in \mathcal{S}$ by descending mass m_i;
Initialize empty result table $\mathcal{T}_5$;
for $t \leftarrow 1$ **to** T **do**
    Initialize empty result table $\mathcal{T}_{4,t}$;
    foreach *recipe* $r \in \mathcal{R}$ **do**
        Initialize heap $H_r \leftarrow \emptyset$, total mass $W_r \leftarrow 0$;
        Initialize element mass sums $A_{r,e} \leftarrow 0 \; \forall e \in \mathcal{E}$;
        repeat
            `// Make multiple passes until no additional item is added`
            foreach *scrap item* $i \in \mathcal{S}$ *in sorted order* **do**
                if TryAdd$(i, H_r, W_r, \{A_{r,e}\}_{e\in\mathcal{E}}, r, M) =$ ***true*** **then**
                    Add i to H_r; update W_r and $\{A_{r,e}\}_{e\in\mathcal{E}}$;
        until ***no item was added to*** H_r;
        $(H_r, W_r, \{A_{r,e}\}) \leftarrow LocalSearch(r, H, W, \{A_e\}, \mathcal{S}, M, I_{\max})$;
        Compute $V_r \leftarrow p_r \cdot W_r$;
        Append (r, H_r, W_r, V_r) in result table $\mathcal{T}_{4,t}$;
    Sort $\mathcal{T}_{4,t}$ in descending order of V_r;
    Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ be the top-ranked row of $\mathcal{T}_{4,t}$;
    foreach *scrap item* $i \in \hat{H}_t$ **do**
        Annotate item i with recipe $\hat{r}_t$ and value $\hat{V}_t$ in $\mathcal{T}_5$;

By combining multi-pass greedy construction with a local search improvement step, the algorithm is able to generate high-quality candidate heaps while still maintaining computational efficiency. The

greedy phase quickly produces feasible solutions, while the local search phase helps escape locally suboptimal configurations and improve the overall mass utilisation and economic value of the resulting heaps.

4.4.6. Multi-start Greedy + Local Search

Algorithm 7: Multi-start Greedy + LocalSearch extends the multi-start greedy heap construction heuristic by incorporating a local search improvement step. The goal of this hybrid method is to combine the exploratory power of randomised greedy construction with the refinement capability of local search. The multi-start phase generates multiple candidate heap configurations by varying the order in which scrap items are considered, while the local search phase attempts to further improve the best configuration by performing targeted item replacements and greedy repacking.

The algorithm takes as input the set of steel recipes $\mathcal{R}$ with their associated economic values p_r and chemical composition bounds $[L_{r,e}, U_{r,e}]$ for each element $e \in \mathcal{E}$. The available scrap items are represented by the set $\mathcal{S}$, where each item i has a mass m_i and elemental composition $x_{i,e}$. In addition, the maximum heap mass M, the number of allocation rounds T, and the number of randomised starting configurations N_{starts} are specified. As in the previous algorithms, the method generates candidate heap tables $\mathcal{T}_{4,t}$ for each allocation round, while the final assignment of scrap items to recipes is recorded in the annotation table $\mathcal{T}_5$.

At the start of the algorithm, the allocation table $\mathcal{T}_5$ is initialised. The algorithm then proceeds through T allocation rounds. At the beginning of each round t, a new candidate heap table $\mathcal{T}_{4,t}$ is created. For every recipe $r \in \mathcal{R}$, the algorithm attempts to construct a high-quality heap through a series of randomised greedy construction attempts.

The heap is initially empty, with $H_r = \emptyset$, total mass $W_r = 0$, and element mass-sums $A_{r,e} = 0$ for all $e \in \mathcal{E}$. For each of the N_{starts} random starts, a temporary copy of the current heap configuration is created, denoted by $(\tilde{H}, \tilde{W}, \{\tilde{A}_e\})$. The scrap items $i \in \mathcal{S}$ are then randomly shuffled to generate a new candidate ordering.

Next, a greedy construction pass is performed over the shuffled scrap items. For each item i, the feasibility routine Function 1: TryAdd is invoked to determine whether the item can be added to the temporary heap without violating the heap capacity constraint or the chemical composition bounds of recipe r. If the item satisfies all constraints, it is added to the temporary heap $\tilde{H}$, and the heap mass $\tilde{W}$ and element mass-sums $\tilde{A}_e$ are updated accordingly.

After all scrap items have been evaluated in the shuffled order, the temporary heap is compared with the current best heap. If the temporary heap achieves a larger total mass ($\tilde{W} > W_r$), it replaces the current best configuration. In this way, the algorithm retains the best heap obtained among the randomised construction attempts.

Once all N_{starts} randomised attempts have been completed, the resulting heap configuration is further refined using the local search procedure described in Function 2: Local Search. The local search explores replacement moves within the heap and performs greedy repacking in order to increase the total heap mass while maintaining feasibility with respect to the capacity and chemical composition constraints.

After the local search procedure terminates, the final heap configuration $(H_r, W_r, \{A_{r,e}\})$ is evaluated. The economic value of the heap is computed as

$$V_r = p_r \cdot W_r,$$

representing the value of producing recipe r with the assembled scrap mixture. The tuple (r, H_r, W_r, V_r) is then appended to the candidate heap table $\mathcal{T}_{4,t}$.

After candidate heaps have been generated for all recipes, the rows of $\mathcal{T}_{4,t}$ are sorted in descending order of their value V_r. The highest-ranked row corresponds to the most valuable feasible heap found in the current allocation round. Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ denote this selected heap.

Finally, the scrap items contained in the selected heap $\hat{H}_t$ are permanently assigned to recipe $\hat{r}_t$. Each item $i \in \hat{H}_t$ is annotated in the allocation table $\mathcal{T}_5$ with the selected recipe $\hat{r}_t$ and its associated value $\hat{V}_t$. These annotations prevent the same scrap items from being used again in later allocation rounds.

By combining randomised greedy construction with local search refinement, the algorithm explores a broader range of heap configurations and improves the quality of the final solutions. The multi-start phase increases diversification by generating multiple candidate packings, while the local search phase intensifies the search around promising solutions to further increase heap utilisation and economic value.

Algorithm 5: Heap Formation and Iterative Allocation (Multi-start Greedy + LocalSearch)

Input: Recipes $\mathcal{R}$ with price p_r and chemical bounds $L_{r,e}, U_{r,e}$; Scrap items $\mathcal{S}$ with masses m_i and compositions $x_{i,e}$; Maximum heap mass M; Number of heaps T; Number of random starts N_{starts}

Output: Result tables $\mathcal{T}_{4,t}$ with scrap item allocation possibilities H_r are ranked according to their value V_r; The scrap items in the recipe with the highest value $i \in \hat{H}_t$ are annotated in $\mathcal{T}_5$ with their recipe $\hat{r}_t$ and value $\hat{V}_t$.

Initialize empty result table $\mathcal{T}_5$;
for $t \leftarrow 1$ **to** T **do**
  Initialize empty result table $\mathcal{T}_{4,t}$;
  foreach *recipe* $r \in \mathcal{R}$ **do**
    Initialize heap $H_r \leftarrow \emptyset$, total mass $W_r \leftarrow 0$;
    Initialize element mass sums $A_{r,e} \leftarrow 0 \; \forall e \in \mathcal{E}$;
    foreach $s \leftarrow 1$ **to** N_{starts} **do**
      $(\tilde{H}, \tilde{W}, \{\tilde{A}_e\}) \leftarrow (H_r, W_r, \{A_{r,e}\})$ `// Make a copy`
      Shuffle scrap items $i \in \mathcal{S}$;
      foreach *scrap item* $i \in \mathcal{S}$ *in shuffled order* **do**
        if TryAdd$(i, \tilde{H}, \tilde{W}, \{\tilde{A}_e\}_{e\in\mathcal{E}}, r, M) =$ ***true*** **then**
          Add i to $\tilde{H}$; update $\tilde{W}$ and $\{\tilde{A}_e\}_{e\in\mathcal{E}}$;
      if $\tilde{W} > W$ **then**
        `// Save best shuffle by highest mass`
        $(H, W, \{A_e\}) \leftarrow (\tilde{H}, \tilde{W}, \{\tilde{A}_e\})$;
    $(H_r, W_r, \{A_{r,e}\}) \leftarrow LocalSearch(r, H, W, \{A_e\}, \mathcal{S}, M, I_{\max})$;
    Compute $V_r \leftarrow p_r \cdot W_r$;
    Append (r, H_r, W_r, V_r) in result table $\mathcal{T}_{4,t}$;
  Sort $\mathcal{T}_{4,t}$ in descending order of V_r;
  Let $(\hat{r}_t, \hat{H}_t, \hat{W}_t, \hat{V}_t)$ be the top-ranked row of $\mathcal{T}_{4,t}$;
  foreach *scrap item* $i \in \hat{H}_t$ **do**
    Annotate item i with recipe $\hat{r}_t$ and value $\hat{V}_t$ in $\mathcal{T}_5$;

4.5. Chapter summary

This chapter investigated which algorithms are suitable for implementing the recipe-based allocation task of the Sorting Decision Layer. The analysis in Section 4.1 showed that the allocation problem has an extremely large solution space. For each heap, the system must evaluate combinations of scrap items against hundreds of possible steel recipes, while respecting mass and chemical composition constraints. Even for moderate buffer sizes, the number of possible item combinations becomes astronomically large. This confirms that exhaustive enumeration is not a practical solution method for the

proposed sorting system.

Section 4.2 describes the computational environment used for implementing and testing the allocation logic. The simulation was developed in VBA within Microsoft Excel to allow transparent inspection of intermediate tables, allocation steps, and constraint checks. Although this environment provides flexibility and traceability during model development, it also reinforces the need for computationally efficient algorithms that can produce feasible solutions without relying on expensive exact optimisation procedures.

The algorithm selection in Section 4.3 showed that exact optimisation methods, such as ILP, MILP, branch-and-bound, and branch-and-cut, are theoretically attractive because they can provide optimality guarantees. However, their computational requirements make them unsuitable as the main solution method for large and repeatedly evaluated scrap allocation problems. Instead, heuristic methods are more appropriate for the proposed Sorting Decision Layer because they can construct feasible solutions quickly and scale better with the number of scrap items, recipes, and allocation rounds.

The heap construction procedures in Section 4.4 therefore focused on greedy and local-search-based heuristics. The Single-run Greedy algorithm provides a fast baseline by constructing candidate heaps in one pass over the available scrap items. The Multi-pass Greedy algorithm extends this by repeatedly scanning the remaining items, increasing the chance of filling unused heap capacity with compatible scrap. The Multi-start Greedy algorithm introduces randomised item orders to explore alternative heap configurations and reduce dependence on one fixed sorting sequence. Finally, the Local Search procedure and the hybrid variants with Local Search attempt to improve already constructed heaps by replacing items and repacking the remaining capacity while maintaining all feasibility constraints.

Together, these sections answer the third sub-research question: “What kind of algorithms are suitable for assigning scrap items within the proposed sorting system, considering solution quality and computational efficiency?” For this problem, suitable algorithms are heuristics that combine fast construction of feasible heaps with limited improvement steps. Greedy methods are suitable because they can efficiently handle large scrap streams and many candidate recipes, while multi-pass and multi-start variants improve solution quality by exploring more feasible combinations. Local Search is suitable as an additional refinement method because it can improve heap utilisation and value without requiring full enumeration of the solution space. Therefore, the candidate algorithms selected for further evaluation are Single-run Greedy, Multi-pass Greedy, Multi-start Greedy, and their Local Search-enhanced variants.

However, this chapter only defined and motivated the candidate algorithms. To determine how well they perform in practice, they must be tested on representative scrap input data and evaluated using quantitative performance indicators. The next chapter, Chapter 5, therefore describes the generation of synthetic scrap data, the simulation setup, the experimental configurations, and the metrics used to compare algorithm performance in terms of scrap conversion, heap utilisation, economic value, and computation time.

5

Simulation data and experiments

The previous chapters developed the Sorting Decision Layer (Chapter 3) and selected candidate algorithms (Chapter 4) capable of constructing recipe-compliant scrap heaps. However, to evaluate whether this approach actually improves scrap allocation, the system must be tested under controlled and repeatable conditions. This chapter therefore translates the proposed allocation model into a simulation framework. It defines the performance indicators used to measure material utilisation, space efficiency, economic value, and computational performance, and it establishes the synthetic scrap dataset and experimental configurations required to compare the candidate algorithms and system parameters.

This chapter addresses the fourth sub-research question: "How can the performance of the proposed scrap sorting system be quantitatively evaluated and tuned for maximal value creation and space efficiency?" First, Section 5.1 defines the key performance indicators used to evaluate the sorting simulations. Next, Section 5.2 describes the experimental setup used to compare candidate algorithms and analyse the effects of buffer size, heap capacity, and number of heaps. Section 5.3 explains how synthetic piece-by-piece scrap composition data is generated from European steel standards to enable reproducible simulation experiments. Finally, the chapter is concluded by summarising its findings and answering the fourth sub-research question in Section 5.4.

5.1. Performance indicators

To evaluate the effectiveness of the proposed scrap sorting system and answer the fourth sub-research question, “How can the performance of the proposed scrap sorting system be quantitatively evaluated and tuned for maximal value creation and space efficiency?”, a set of quantitative performance indicators is defined. These indicators are chosen to measure material efficiency, economic performance, and operational feasibility, and are used in Chapter 6 to compare system configurations and candidate algorithms:

The performance of the sorting and allocation system is evaluated using several performance indicators that quantify material utilisation, economic output, and system productivity. These indicators allow different algorithmic configurations to be compared in a consistent and interpretable manner.

- **Item conversion (%)** is defined as the fraction of individual scrap items that are successfully assigned to a heap relative to the total number of scrap items entering the system. Let $\mathcal{S}$ denote the set of all scrap items and let $\sum_{t=1}^{T} |\hat{H}_t|$ denote the set of scrap items that are assigned to the selected heaps. The item conversion is defined as

$$\textit{Item conversion} = \frac{\sum_{t=1}^{T} |\hat{H}_t|}{|\mathcal{S}|} \times 100\%. \tag{5.1}$$

  This indicator measures how effectively the algorithm utilises the available scrap items. A higher item conversion indicates that fewer scrap pieces remain unused, which is desirable because unused items represent potential material waste.

- **Mass conversion (%)** measures the fraction of total incoming scrap mass that is successfully allocated to valid heaps. Let m_i denote the mass of scrap item i. The mass conversion is defined as

$$\textit{Mass conversion} = \frac{\sum_{t=1}^{T} \hat{W}_t}{\sum_{i \in \mathcal{S}} m_i} \times 100\%. \tag{5.2}$$

  This indicator focuses on material utilisation in terms of mass rather than item count. It is particularly relevant for industrial scrap handling systems because large scrap pieces may represent a significant portion of the total material flow.

- **Heap utilisation (%)** represents the fraction of the maximum allowable heap capacity that is effectively used. Given a maximum heap capacity M and T constructed heaps, the heap utilisation is defined as

$$\textit{Heap utilization} = \frac{\sum_{t=1}^{T} \hat{W}_t}{T \cdot M} \times 100\%. \tag{5.3}$$

  This indicator reflects how efficiently the available heap capacity is filled. Higher utilisation indicates that the algorithms are able to construct heaps that are close to the maximum allowable mass, thereby improving production efficiency.

- **Value (€)** quantifies the economic output of the sorting system. The value of a heap is determined by multiplying the mass of the heap by the market price of the corresponding steel recipe. The total value generated by the system is therefore

$$\textit{Value} = \sum_{t=1}^{T} p_{\hat{r}_t} \hat{W}_t, \tag{5.4}$$

  where $\hat{r}_t$ denotes the recipe associated with heap t. This indicator directly reflects the economic objective of the system and therefore serves as the primary measure for comparing algorithmic performance.

In the context of the proposed sorting system, the Equation 5.4: Value indicator is considered the most important performance metric. The primary objective of the decision-support system is to maximise the economic value of the produced steel heaps by assigning scrap items to the most valuable feasible recipes. Consequently, the value metric directly reflects the effectiveness of the allocation algorithms and serves as the main criterion for comparing different system configurations.

The second most important indicator is Equation 5.3: Heap utilisation, which captures the space-efficiency of the heap formation process. Efficient use of the available heap capacity is important in practice because it reflects how effectively the system converts sorting results into usable furnace charges. High heap utilisation indicates that the algorithms are able to construct heaps that approach the maximum allowable mass, thereby minimising unused heap capacity.

The Equation 5.2: Mass conversion indicator is relevant in situations where no explicit economic value can be assigned to individual scrap items or groups of items. In such cases, maximising the fraction of total scrap mass that can be incorporated into valid heaps provides a meaningful measure of material efficiency.

The Equation 5.1: Item conversion indicator provides additional insight into the allocation decisions made by the system. While it is less directly related to economic performance, it helps to interpret how the algorithm distributes scrap items across heaps and how many items remain unused.

Finally, although Throughput (kg/hour) is generally an important performance indicator in industrial sorting systems, it is not listed in this thesis because of the limited relevance in the decision-support layer studied in this thesis. In practice, the throughput of a sensor-based scrap sorting system is primarily determined by the physical sorting equipment rather than the computational time of the allocation algorithm. As discussed in Section 2.3, the sorting hardware represents the dominant bottleneck in the system. Therefore, the computational throughput of the proposed algorithms is not expected to significantly constrain the overall system performance.

5.2. Simulation experiments

This section describes the design and setup of the simulation experiments used to evaluate the proposed scrap sorting system. The experiments aim to analyse the performance of the sorting decision layer under varying system configurations and to identify parameter settings that maximise value creation and space efficiency.

First, the key simulation model parameters are introduced in Section 5.2.1, which define the operational constraints and influence the search space explored by the allocation model. Next, a comparison of candidate algorithms is conducted to select the most suitable approach for the subsequent experiments in Section 5.2.2.

The remaining experiments systematically evaluate the effect of key parameters, including buffer size, number of heaps, and heap capacity, on system performance. These experiments are conducted using a consistent baseline configuration and are assessed using the performance indicators defined in Section 5.1. Finally, the trade-off between the two most important performance indicators, economic performance and space efficiency, is analysed by the last experiment in Section 5.2.7.

5.2.1. Simulation model parameters

The simulation model contains several parameters that define both the operational constraints of the simulated sorting system and the search space explored by the allocation algorithms. These parameters determine how many scrap items are available for allocation, how many recipe-specific heaps can be formed, and how much scrap mass each heap can contain. In the simulation experiments, these parameters are systematically varied to evaluate their influence on material conversion, heap utilisation, economic value, and computational performance. The resulting trends are used in Chapter 6 to identify suitable parameter settings and to assess how the proposed Sorting Decision Layer performs under different system configurations.

- **Buffer size (number of scrap items)** determines how many scrap items are simultaneously available to the allocation algorithm in a single simulation round. Let $\mathcal{S}$ denote the set of scrap items currently present in the buffer. The buffer size is defined as

$$|\mathcal{S}| = B, \tag{5.5}$$

where B represents the maximum number of scrap items that can be stored in the buffer shown in Figure 3.1. The buffer size influences the size of the decision space available to the algorithm. A larger buffer increases the number of possible item combinations that can be assigned to heaps, potentially improving solution quality, but also increases the computational complexity of the allocation problem. This parameter is therefore tuned to balance solution quality and computational tractability while reflecting realistic buffer capacities in industrial sorting systems.

- **Number of heaps** represents the maximum number of heaps that can be constructed in a single allocation round. Let T denote the number of heaps available to the algorithm, with each selected heap represented by $\hat{H}_t$ for $t = 1, \ldots, T$. The constraint can be expressed as

$$t \in \{1, \ldots, T\}. \tag{5.6}$$

This parameter limits the number of candidate steel grades that can be produced simultaneously in the model. Increasing T allows the system to consider a wider range of recipes and therefore increases allocation flexibility. However, a larger number of heaps also increases the search space of the optimisation problem. The parameter T is therefore tuned to reflect realistic operational constraints in scrap processing facilities.

- **Heap capacity (kg)** defines the maximum allowable scrap mass that may be assigned to a single heap. Let W_r denote the total mass of heap H_r. The heap capacity constraint is expressed as

$$W_r \leq M. \tag{5.7}$$

Here, M represents the maximum allowable heap mass in kilograms. This constraint ensures that the simulated heaps remain consistent with practical production limitations such as furnace charge size or handling capacity. The value of M also affects the packing behaviour of the allocation algorithms: smaller capacities force the algorithm to construct more compact heaps, while larger capacities increase the flexibility to combine scrap items. The parameter is therefore tuned to represent realistic batch sizes used in steel production.

Together, the performance indicators defined in Section 5.1 and the simulation parameters defined above form the basis for the experimental evaluation of the proposed Sorting Decision Layer. The performance indicators determine how each simulation result is assessed, while the parameters determine the operational conditions under which the allocation algorithms are tested. The following experiments therefore systematically combine these two components: first by selecting a suitable baseline algorithm, then by analysing the effect of buffer size, and finally by evaluating how different heap capacities and numbers of heaps influence mass conversion, heap utilisation, economic value, and the trade-off between value creation and space efficiency.

5.2.2. Comparison of candidate algorithms

Before experiments can be performed, an algorithm has to be selected from the candidate algorithms proposed in Chapter 4 based on the performance indicators discussed in Section 5.1.

For this preliminary experiment, all candidate algorithms have to process the same batch of 1000 scrap pieces generated according to the scrap data generation procedure proposed in Section 5.3. To prevent excessive compute times, every candidate algorithm evaluates only the first most valuable heap without a constraint on heap capacity.

The results will be displayed in a table with columns: 'Algorithm', 'Item Conversion (%)', 'Mass Conversion (%)', 'Value (€)', and 'Compute Time (ms)', and the tested algorithms: 'Single-run Greedy', 'Multi-pass Greedy', 'Multi-pass Greedy + local search', 'Multi-start Greedy', and 'Multi-start Greedy + local search' in rows.

5.2.3. Assessment of buffer size effects

To select the optimal number of heaps and heap capacity, you first have to select the right amount of scrap items fed into the systems per cycle: Equation 6.6: Buffer size. A buffer size chosen too small reduces the ability to assemble effective scrap mixtures. A buffer size chosen too large reduces the space efficiency of the system.

For this experiment, the numerical simulation will sort buffer sizes ranging between 0 and 2000 scrap pieces, each generated according to the procedure outlined in Section 5.3. The performance of the first heap most valuable heap (without heap capacity constraint), will be measured for each buffer size according to the performance indicators discussed in Section 5.1.

Results will be displayed in a table with columns: 'Buffer size (number of scrap pieces)', 'Item Conversion (%)', 'Mass Conversion (%)', and 'Value (€)', with the tested buffer sizes in rows. Next to the table, a figure will display a scatterplot of the 'Mass Conversion (%)' against the 'Buffer size (number of scrap pieces)', including a polynomial trend-line for added clarity.

5.2.4. Mass conversion performance

After determining a stable buffer size, different amounts of heaps: Equation 6.7: Number of heaps, can be tested against different heap sizes: Equation 6.8: Heap capacity. The performance of these different heap configurations is measured in three different ways, the first: Equation 6.2: Mass conversion is displayed in this experiment.

For this experiment and the next experiments outlined in subsections 5.2.5, 5.2.6, and 5.2.7, the same baseline configuration is used. This baseline configuration consists of the same batch of scrap items generated according to Section 5.3, with the number of scrap items identified in Section 6.2 as a stable buffer size. The mass conversion performance of all different combinations between heap sizes 0 kg and 20 000 kg, and 0 and 12 number of heaps will be explored in this experiment.

Results will be cumulatively displayed (adding the results of the current and previous evaluated heaps together) in a table with columns: 'Heap Size (kg)' representing the capacity per heap, and 'Number of heaps' available for use in the simulation for that particular configuration to use. For extra clarity, results in the table will be shaded from light to dark, from low to high. Next to the table, a figure will display a plot of the cumulative 'Mass Conversion (%)' against the 'Heap capacity (kg)' for different number of heaps to view the same data presented in the table from another perspective for potentially more insight.

5.2.5. Heap utilisation performance

The second performance metric measured for the same heap configurations outlined for experiment 'Mass conversion performance under baseline configuration' in subsection 5.2.4 is the amount of heap capacity used: Equation 6.3: Heap utilisation.

Results will be cumulatively displayed (adding the results of the current and previous evaluated heaps together) in a table with columns: 'Heap Size (kg)' representing the capacity per heap, and 'Number of heaps' available for use in the simulation for that particular configuration to use. For extra clarity, results in the table will be shaded from light to dark, from low to high. Next to the table, a figure will display a plot of the cumulative 'Heap utilisation (%)' against the 'Heap capacity (kg)' for different number of heaps to view the same data presented in the table from another perspective, for potentially more insight.

5.2.6. Economic performance

The third performance metric measured for the same heap configurations outlined for experiment 'Mass conversion performance under baseline configuration' in subsection 5.2.4 is the amount of heap capacity used: Equation 6.4: Value.

Results will be cumulatively displayed (adding the results of the current and previous evaluated heaps together) in a table with columns: 'Heap Size (kg)' representing the capacity per heap, and 'Number of heaps' available for use in the simulation for that particular configuration to use. For extra clarity, results in the table will be shaded from light to dark from low to high. Next to the table, a figure will display a plot of the cumulative 'Value (€)' against the 'Heap capacity (kg)' for different number of heaps to view the same data presented in the table from another perspective, potentially for more insight.

5.2.7. Performance trade-offs between heap utilisation and revenue

This last experiment combines the results from '5.2.5 Mass conversion performance under baseline configuration' and from '5.2.6 Economic performance under baseline configuration', by multiplying the 'Heap utilization (%)' with the 'Value (€)' of each corresponding heap configuration. This way, an optimum can be identified that maximises the created value against the space efficiency of the tested heap configurations.

Results will be displayed in a table with columns: 'Heap Size (kg)' representing the capacity per heap, and 'Number of heaps' available for use in the simulation for that particular configuration to use. For extra clarity, results in the table will be shaded from light to dark from low to high. Next to the table, a figure will display a plot of the 'Value (€) x Heap utilisation (%)' against the 'Heap capacity (kg)' for different number of heaps to view the same data presented in the table from another perspective, for potentially more insight.

The composition of the item-conversion rate and mass-conversion rate results of the heap configuration with the highest combined score ('Value (€) x Heap utilization (%)') will be displayed in two pie charts, where the item% and mass% of each steel recipe is displayed relative to the total item count and total scrap mass of the imputed scrap data (the buffer size). Also, a plot of every heap with its assigned recipe will be provided, with the element mass-percentage set out against all relevant elements according to the EU Steel standards, including the element limit ranges corresponding to the assigned recipe op that heap. To give a complete picture of the scrap allocation, also the elemental composition is provided of the leftover scrap elements that could not be assigned to one of the heaps.

These experiments define how the proposed Sorting Decision Layer is evaluated under different algorithmic and operational conditions. The sequence of experiments first selects a suitable baseline algorithm, then determines a stable buffer size, and finally analyses how heap capacity and the number of heaps influence mass conversion, heap utilisation, economic value, and the combined value-utilisation trade-off. However, all these experiments require a consistent input dataset containing the mass and elemental composition of individual scrap items. Since large-scale public datasets with piece-level scrap compositions are not readily available, the next section describes how synthetic scrap data is generated to provide a reproducible input for the simulation model.

5.3. Scrap Generation

As piece-by-piece scrap compositional data is not available for this research, synthetic scrap data is generated based on European steel standards, as described in subsection 5.3.2. The data is combined and prepared according to the data framework and assumptions outlined in subsection 5.3.1. Finally, the synthetic scrap data is generated following the procedure described in subsection 5.3.3.

5.3.1. Scrap composition and data assumptions

The scrap input data used in this study is synthetically generated to provide a controlled and transparent basis for evaluating the proposed sorting strategy. Scrap items are assumed to originate from previously produced steel grades that comply with established European Steel Standards. An overview of the standards used to define the underlying steel compositions is given in Table 5.1. These standards cover a broad range of steel families, including structural steels, pressure vessel steels, stainless steels, and tool steels, thereby representing a diverse and realistic scrap pool.

For each standard, representative steel grades were translated into a steel composition table containing minimum and maximum mass fractions for relevant alloying and residual elements. Where multiple configurations exist within a standard, the thinnest and/or strongest configuration was chosen. This choice reflects conservative composition limits, as high-strength or thin-gauge products typically have the tightest chemical constraints. Using these limiting cases ensures that the resulting recipes are well-defined and restrictive, which is ideal for testing the feasibility of recipe-based scrap sorting.

To maintain clarity, the composition data is limited to direct elemental bounds. Composite or derived constraints, such as carbon equivalent values (CEV), were deliberately excluded. While such rules are important in industrial steelmaking, they introduce complexity that is not relevant in this initial system design stage. By relying solely on explicit minimum and maximum elemental contents, each recipe constraint remains directly applicable in the numerical simulation.

Individual scrap items are generated by sampling elemental compositions within the specified bounds of their parent steel grade and assigned a random mass between 1 and 200 kg. Scrap pieces are treated as chemically uniform and indivisible units. Effects such as surface contamination, oxidation, coating layers, or sensor measurement uncertainty are not explicitly modelled. These simplifications allow the study to focus on the system-level behaviour of recipe-based sorting rather than sensor-level inaccuracies.

5.3.2. Data source

The synthetic scrap dataset is based on elemental composition data provided by het Nederlands Normalisatie Instituut (NEN), which defines the chemical composition limits for European steel grades. The elemental bounds of the steel grades used in this study are listed in Table 5.1. From these standards, an input recipe table $\mathcal{T}_1$ is assembled with columns for the steel symbol and the minimum and maximum allowable mass fraction of each element for all 315 recipes. Because publicly available price data for individual EU steel grades were not available, synthetic prices between €0.50/kg and €5.00/kg were assigned to all listed steel grades. This range was used as an assumed representative price range for EU steel grades. A screenshot illustrating the Excel implementation of the input recipe table $\mathcal{T}_1$ is shown in Figure B.1.

Standard	**Steel type**
[38] NEN-EN 10025-2:2019	Non-alloy structural steel
[39] NEN-EN 10025-3:2019	Fine-grain structural steel
[40] NEN-EN 10025-4:2019 + A1:2022	Fine-grain structural steel
[41] NEN-EN 10025-5:2019	Structural steel with improved atmospheric corrosion resistance
[42] NEN-EN 10025-6:2019 + A1:2022	High-strength quenched and tempered structural steel
[43] NEN-EN 10028-2:2017	Pressure vessel steel
[44] NEN-EN 10028-3:2017	Fine-grain pressure vessel steel
[45] NEN-EN 10028-4:2017	Nickel-alloyed pressure vessel steel
[46] NEN-EN 10028-5:2017	Thermomechanically rolled pressure vessel steel
[47] NEN-EN 10028-6:2017	Quenched and tempered pressure vessel steel
[48] NEN-EN 10028-7:2016	Stainless steels for pressure purposes
[49] NEN-EN 10088-1:2023	List of stainless steels
[50] NEN-EN 10088-2:2024	Stainless steel sheet/plate and strip
[51] NEN-EN 10088-3:2024	Stainless steel bars, rods, wire, sections and bright products
[52] NEN-EN 10088-4:2009	Stainless steels for structural applications
[53] NEN-EN 10088-5:2009	Stainless steel bars, rods and wire for construction purposes
[54] NEN-EN-ISO 4957:2018	Tool steels

Table 5.1: European steel standards used for the numerical simulation.

5.3.3. Data generation model

The per-item scrap steel data for the simulation model outlined in Chapter 3 is produced according to the following steps in Excel:

1. A number of recipe rows equal to the desired number of scrap items is sampled with replacement from the steel recipe composition table $\mathcal{T}_1$ and copied into a new table $\mathcal{T}_2$. This simulates the variety of steel grades present in the incoming scrap stream.
2. For each element in each row of $\mathcal{T}_2$, a value is randomly sampled between the corresponding minimum and maximum composition limits and stored in a new table $\mathcal{T}_3$ on the same row. This represents the elemental composition variation of scrap steel within the bounds of European steel recipes.
3. A random mass between 1 and 200 kg is assigned to each row of $\mathcal{T}_3$, representing an assumed typical mass range for individual scrap items.
4. A scrap ID is assigned to each row of $\mathcal{T}_3$ by numbering the rows in ascending order. This ID is used to assign the scrap items to the different heaps.

Screenshots illustrating the Excel implementation of the input recipe table $\mathcal{T}_1$, the candidate scrap table $\mathcal{T}_2$, and the generated scrap-item table $\mathcal{T}_3$ are shown in Figures B.1 to B.3, respectively.

5.3.4. Data generation pseudocode

Procedure 1: Scrap Data Generation describes the procedure used to generate a synthetic set of scrap items from the available recipe data. Because real scrap item data with detailed elemental compositions and masses were not directly available for all experiments, a simulated scrap dataset was constructed based on the chemical composition ranges of the steel recipes. The purpose of this procedure is to create a diverse set of scrap items that reflects the composition space of the recipe database while allowing controlled variation in both chemistry and mass.

The algorithm takes as input the recipe table $\mathcal{T}_1$, which contains one row for each recipe $r \in \mathcal{R}$ and, for every tracked element $e \in \mathcal{E}$, the corresponding minimum and maximum composition values stored in the columns "e % min." and "e % max.". In addition, a target sample size N is specified, representing the number of synthetic scrap items to be generated. The output of the procedure is a scrap item set $\mathcal{S}$, stored in output table $\mathcal{T}_3$, in which each item i is characterised by a mass m_i and elemental composition values $x_{i,e}$.

The generation process consists of two main steps. In the first step, rows are sampled from the recipe table with replacement. An empty candidate scrap table $\mathcal{T}_2$ is initialised, after which N row indices are drawn independently from a discrete uniform distribution over the set of recipe rows,

$$j_i \sim \mathrm{Uniform}\{1, \ldots, |\mathcal{R}|\}, \qquad i = 1, \ldots, N.$$

For each sampled index j_i, the corresponding row from $\mathcal{T}_1$ is copied into row i of $\mathcal{T}_2$. Because sampling is performed with replacement, the same recipe may be selected multiple times. This allows the generated scrap dataset to contain multiple synthetic items derived from the same recipe template.

In the second step, the elemental compositions of the sampled rows are randomised within their specified lower and upper bounds. An empty output table $\mathcal{T}_3$ and an empty scrap set $\mathcal{S}$ are initialised. For each sampled row i, the algorithm loops over all tracked elements $e \in \mathcal{E}$. The lower and upper bounds $L_{i,e}$ and $U_{i,e}$ are read from the corresponding "e % min." and "e % max." columns. If both values are numeric, the elemental composition of scrap item i for element e is generated as

$$x_{i,e} \sim \mathrm{Uniform}(L_{i,e}, U_{i,e}).$$

If one or both bounds are not available, the corresponding elemental composition is set to zero. In this way, each sampled recipe row is transformed into a unique synthetic composition vector while remaining within the chemical range of the originating recipe.

After the elemental composition values have been generated, a random mass is assigned to the scrap item. The item mass is sampled from a continuous uniform distribution,

$$m_i \sim \mathrm{Uniform}(m_{\min}, m_{\max}),$$

where in the implementation $m_{\min} = 0.1$ kg and $m_{\max} = 200$ kg. This introduces variation in scrap item size and ensures that the generated dataset contains both small and large scrap pieces.

Once the composition vector $\{x_{i,e}\}_{e\in\mathcal{E}}$ and mass m_i have been generated, the synthetic scrap item is added to the scrap set $\mathcal{S}$ and appended to the output table $\mathcal{T}_3$. Repeating this procedure for all $i = 1, \ldots, N$ yields a complete synthetic scrap dataset.

The resulting dataset provides a computationally convenient representation of varying scrap input material for the allocation heuristics. By combining recipe-based sampling with random composition generation within the specified min-max bounds, the procedure preserves the general chemical characteristics of the recipe database while introducing sufficient variability for simulation performance evaluation.

Procedure 1: Scrap Data Generation (Sampling + Random Composition)

Input: Recipe table $\mathcal{T}_1$ ($\mathcal{R}$ rows with "e % min." and "e % max." columns per element $e \in \mathcal{E}$ for every recipe $r \in \mathcal{R}$); target sample size N

Output: Scrap item set $\mathcal{S}$ with masses m_i, and compositions $x_{i,e}$ in table $\mathcal{T}_3$

Step 1: Sample rows with replacement;
Initialize empty candidate scrap table $\mathcal{T}_2$;
for $i \leftarrow 1$ **to** N **do**
    Draw index $j_i \sim$ Uniform$\{1, \ldots, \mathcal{R}\}$;
    Copy row j_i from $\mathcal{T}_1$ into row i of table $\mathcal{T}_2$;

Step 2: Generate random compositions between min/max bounds;
Initialize empty output table $\mathcal{T}_3$;
Initialize empty scrap set $\mathcal{S} \leftarrow \emptyset$;
for $i \leftarrow 1$ **to** N **do**
    foreach $e \in \mathcal{E}$ **do**
        Read $L_{i,e} \leftarrow$ value in row i of column "e % min.";
        Read $U_{i,e} \leftarrow$ value in row i of column "e % max.";
        if $L_{i,e}$ *and* $U_{i,e}$ *are numeric* **then**
            Sample $x_{i,e} \sim$ Uniform$(L_{i,e}, U_{i,e})$;
        else
            `// IF not available`
            Set $x_{i,e} \leftarrow 0$;
        Add $x_{i,e}$ to $\mathcal{S}$
    Sample mass $m_i \sim$ Uniform$(m_{\min}, m_{\max})$; // $m_{\min} = 0.1$, $m_{\max} = 200$ kg
    Add m_i to $\mathcal{S}$
    Append $\left(m_i, \{x_{i,e}\}_{e\in\mathcal{E}}\right)$ to output table $\mathcal{T}_3$;
return $\mathcal{S}$;

5.4. Chapter summary

This chapter defined the simulation framework used to quantitatively evaluate the proposed scrap sorting system. Section 5.1 introduced the key performance indicators used to assess the allocation results. Item conversion and mass conversion measure how much of the incoming scrap stream is successfully assigned to valid heaps. Heap utilisation measures how efficiently the available heap capacity is used, while the value indicator measures the economic output of the selected recipe-based heaps.

Together, these indicators make it possible to evaluate the system from both a material-efficiency and economic-performance perspective.

Section 5.2 then describes the experimental setup used to evaluate and tune the Sorting Decision Layer. The simulation parameters define the operational conditions under which the allocation algorithms are tested. Buffer size determines how many scrap items are available to the algorithm at once, the number of heaps determines how many recipe-specific outputs can be formed, and heap capacity limits the maximum mass assigned to each heap. These parameters directly influence both the search space and the practical performance of the sorting system.

The chapter also defined the sequence of simulation experiments. First, the candidate algorithms from Chapter 4 are compared to select a suitable algorithm for further testing. Next, buffer size is varied to identify a stable amount of available scrap items for each allocation cycle. After that, combinations of heap capacity and number of heaps are evaluated under a baseline configuration. These experiments measure mass conversion, heap utilisation, and economic value separately, after which a combined value-utilisation metric is used to identify configurations that balance revenue generation with efficient use of sorting capacity.

Finally, Section 5.3 describes how synthetic piece-by-piece scrap data is generated from European steel standards. Because publicly available industrial datasets with item-level scrap mass and elemental composition are not available at sufficient scale, the simulation uses generated scrap items based on steel recipe composition ranges. This enables controlled and reproducible testing of recipe-based allocation strategies. The generated data provides each scrap item with a mass, composition vector, and item identifier, making it suitable for evaluating heap formation algorithms under consistent conditions.

Together, these sections answer the fourth sub-research question: “How can the performance of the proposed scrap sorting system be quantitatively evaluated and tuned for maximal value creation and space efficiency?” The performance of the system can be evaluated by applying the proposed allocation algorithms to controlled synthetic scrap datasets and measuring item conversion, mass conversion, heap utilisation, and economic value. The system can be tuned by systematically varying buffer size, heap capacity, and number of heaps, and by comparing how these parameters affect both the value generated and the utilisation of available heap space.

This chapter therefore establishes the experimental basis for evaluating the proposed Sorting Decision Layer. The next chapter, Chapter 6, applies these experiments and presents the results of algorithm comparison, buffer-size analysis, baseline performance evaluation, and value-utilisation trade-off. These results are used to determine which algorithm and system configuration provide the most effective balance between value creation and space efficiency.

6

Results

The previous chapter (Chapter 5) defined the simulation data, performance indicators, and experimental setup used to evaluate the proposed Sorting Decision Layer. This chapter applies that framework to quantify how the recipe-based scrap allocation system performs under different algorithmic and operational configurations. The results show how algorithm choice, buffer size, heap capacity, and number of heaps influence material conversion, heap utilisation, and economic value creation. In doing so, the chapter moves from the design and simulation setup of the proposed system to its measured performance and identifies the configuration that provides the most balanced trade-off between value creation and space efficiency.

This chapter presents the results of the simulation experiments outlined in Section 5.2 for the proposed sorting system described in Chapter 3. First, the candidate algorithms are compared in Section 6.1, after which the effect of buffer size is assessed in Section 6.2. Different system configurations are then evaluated in terms of mass conversion in Section 6.3, heap utilisation in Section 6.4, economic value in Section 6.5, and the trade-off between heap utilisation and revenue in Section 6.6. The chapter concludes by summarising the results and answering the fifth sub-research question: "How does the proposed scrap sorting system perform under varying system configurations?" in Section 6.7.

6.1. Comparison of candidate algorithms

Table 6.1 presents a preliminary quantitative comparison of the candidate greedy-based algorithms using four performance indicators defined in Section 5.1: item conversion, mass conversion, economic value, and computation time. The purpose of this experiment is not yet to identify the best system configuration, but to select a suitable baseline algorithm for the remaining configuration experiments. The selected algorithm should therefore provide a good compromise between solution quality and computational efficiency, because it will be applied repeatedly across different buffer sizes, heap capacities, and numbers of heaps.

The single-run greedy algorithm performs the weakest on all quality-related metrics. It converts 33% of the available scrap items and 24% of the total scrap mass, resulting in a total value of €84 168. Although its computation time is the lowest at 23 359 ms, its value is 52% lower than the multi-pass greedy result and 54% lower than the best multi-start result. This indicates that a single greedy pass is too limited for the proposed allocation problem.

Introducing multiple passes substantially improves performance. The multi-pass greedy algorithm increases item conversion from 33% to 50% and mass conversion from 24% to 49%, corresponding to improvements of 17 and 25 percentage points, respectively. At the same time, the total value increases by 108%, from €84 168 to €175 218. Adding local search to the multi-pass greedy algorithm does not further improve item conversion, mass conversion, or value in this experiment. Instead, computation time increases from 65 223 ms to 114 316 ms, providing no additional benefit.

The multi-start greedy algorithms achieve the highest conversion rates, reaching 61% item conversion and 51% mass conversion. Compared with multi-pass greedy, this represents an additional 11 percentage points in item conversion and 2 percentage points in mass conversion. However, the economic value increases by only 4.5%, from €175 218 to €183 142. This improvement comes at a disproportionately high computational cost: runtime increases from 65 223 ms to 636 020 ms, approximately 9.8 times higher. Adding local search further increases runtime to 846 035 ms while slightly reducing the resulting value.

Based on these KPIs, the multi-start greedy algorithm gives the highest absolute solution quality, but it is not the most balanced choice for the remainder of the thesis. Its small improvement in mass conversion and value does not justify the large increase in computation time, especially because the following experiments require repeated simulations over many parameter combinations. In contrast, the multi-pass greedy algorithm captures 95.7% of the best observed value while requiring only 10.3% of the computation time of the best-performing multi-start algorithm. It also substantially outperforms the single-run greedy algorithm on item conversion, mass conversion, and value.

For this reason, the multi-pass greedy algorithm is selected as the baseline algorithm for the subsequent experiments. The remaining results in Chapter 6 therefore compare different system configurations using a fixed algorithmic approach. The later baseline configuration, consisting of a selected buffer size, heap capacity, and number of heaps, is separate from the baseline algorithm selected in this section.

Algorithm	Item Coversion	Mass Conversion	Value	Compute Time
Single-run Greedy	33%	24%	€ 84.168,21	23359 ms
Multi-pass Greedy	50%	49%	€ 175.218,18	65223 ms
Multi-pass Greedy + local search	50%	49%	€ 175.218,18	114316 ms
Multi-start Greedy	61%	51%	€ 183.142,54	636020 ms
Multi-start Greedy + local search	61%	51%	€ 181.314,47	846035 ms

Table 6.1: Performance comparison of candidate algorithms.

6.2. Assessment of buffer size effects

Table 6.2 and Figure 6.1 illustrate the influence of buffer size on the performance of the heap formation process. The buffer represents the number of scrap items simultaneously available for allocation, and therefore determines the number of feasible scrap combinations that the algorithm can explore.

At small buffer sizes, both item conversion and mass conversion increase rapidly as the buffer grows. For example, increasing the buffer size from 50 to 300 scrap items raises item conversion from 18% to 48% and mass conversion from 16% to 38%, corresponding to increases of 30 and 22 percentage points, respectively. Over the same interval, the total value increases from €1 972 to €41 863. This steep improvement occurs because a larger buffer increases the combinatorial flexibility of the allocation process, making it more likely that feasible scrap combinations satisfying the recipe constraints can be formed.

Beyond approximately 600-700 scrap items, the improvement rate decreases, and the performance metrics begin to fluctuate around a plateau. From 600 to 2 000 items, mass conversion remains within a range of 40-55%, while item conversion remains within 46-59%. As shown in Figure 6.1, the fitted trendline therefore indicates diminishing gains rather than a continued proportional increase. Increasing the buffer further mainly changes the distribution of feasible combinations, but does not systematically improve conversion efficiency.

Although the highest economic value is obtained at the largest tested buffer size (2 000 items), this increase is partly caused by the larger amount of available material. In conversion terms, the gain is more limited: increasing the buffer from 1 000 to 2 000 items raises item conversion from 48% to 50% and mass conversion from 43% to 50%, corresponding to increases of only 2 and 7 percentage points. These improvements are relatively small compared to the doubling of the buffer size and the associated computational effort required to evaluate the expanded solution space.

Considering this trade-off between solution quality and computational complexity, a buffer size of 1000 scrap items is selected for the baseline configuration used in the subsequent experiments. At this level, the performance metrics have already entered the plateau region observed in Figure 6.1, meaning that most of the attainable conversion performance is captured while maintaining manageable computational requirements. Larger buffers can still increase absolute value, but provide only limited additional conversion efficiency.

Therefore, a buffer size of 1 000 scrap items is adopted as the baseline configuration for the experiments presented in Section 6.3, Section 6.4, and Section 6.5.

Buffer Size	Item Conversion	Mass Conversion	Value
0	0%	0%	€ 0
50	18%	16%	€ 1.972,44
100	21%	18%	€ 6.038,11
200	33%	26%	€ 19.054,07
300	48%	38%	€ 41.863,43
400	44%	36%	€ 56.874,63
500	48%	41%	€ 71.588,44
600	59%	55%	€ 120.233,82
700	58%	54%	€ 130.721,54
800	46%	40%	€ 119.647,30
900	56%	49%	€ 161.149,69
1000	48%	43%	€ 154.507,18
1100	53%	48%	€ 190.778,73
1200	57%	53%	€ 234.982,63
1300	49%	46%	€ 221.797,24
1400	52%	50%	€ 251.424,24
1500	53%	50%	€ 274.833,39
1600	48%	45%	€ 261.004,99
1700	53%	48%	€ 296.461,25
1800	56%	53%	€ 341.927,08
1900	49%	46%	€ 317.580,47
2000	50%	50%	€ 358.066,29

Table 6.2: Performance comparison of the most valuable heap across different buffer sizes.

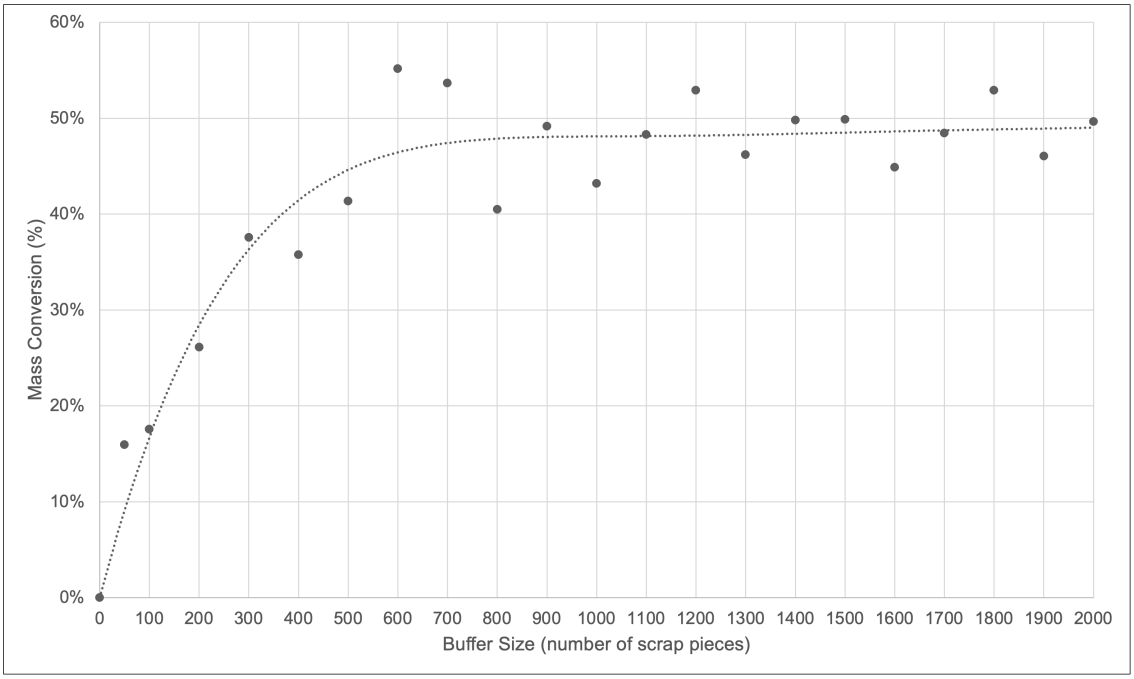


Figure 6.1: Cumulative mass conversion as a function of buffer size. The scatter points show the observed simulation results, while the dashed trendline highlights the diminishing performance gains for larger buffers. Data from Table 6.2.

6.3. Mass conversion performance

Table 6.3 and Figure 6.2 present the cumulative mass conversion achieved for different heap capacities and numbers of heaps under the baseline configuration. Together, these results illustrate how the available heap capacity and the number of simultaneous heaps influence the ability of the allocation algorithm to convert scrap mass into valid steel recipes.

At small heap capacities, mass conversion remains limited because only a small amount of scrap can be assigned to each heap. For example, with a heap capacity of 1 000 kg, conversion ranges from 1% with a single heap to 12% with twelve heaps. Even when the number of heaps is increased from one to twelve, the absolute gain is therefore limited to 11 percentage points. As heap capacity increases, conversion rises rapidly because larger heaps allow more scrap combinations that satisfy the chemical recipe constraints.

This trend is clearly visible in both Table 6.3 and Figure 6.2. For instance, increasing heap capacity from 2 000 kg to 6 000 kg raises the conversion for six heaps from 12% to 37%, corresponding to an increase of 25 percentage points. For twelve heaps, conversion increases from 25% to 68% over the same capacity range, an increase of 43 percentage points. This steep increase indicates that larger heap capacities significantly improve the combinatorial flexibility of the allocation process, enabling more scrap items to be combined into feasible heaps.

Beyond approximately 8 000-10 000 kg, the rate of improvement begins to decrease, especially for configurations with larger numbers of heaps. For example, with twelve heaps, conversion increases from 80% at 8 000 kg to 83% at 10000 kg, and then to 92% at 20 000 kg. This means that the final doubling of heap capacity from 10 000 kg to 20 000 kg adds only 9 percentage points. Similar saturation behaviour is observed for other high-heap configurations, indicating that many suitable scrap combinations have already been identified at these capacities.

Increasing the number of heaps consistently improves mass conversion because it allows more steel recipes to be produced simultaneously. However, the marginal benefit of additional heaps diminishes at larger capacities. For example, at 6 000 kg, increasing from eight to twelve heaps raises conversion from 48% to 68%, a gain of 20 percentage points. At 10 000 kg this difference decreases to 11 percentage points, and at 20 000 kg it decreases further to only 4 percentage points. The scatter points and fitted trendlines in Figure 6.2 therefore show convergence at higher heap capacities, rather than a constant benefit from adding more heaps.

These results show that heap capacity and number of heaps should not be interpreted independently. Heap capacity determines how much material can be assigned to each selected recipe, while the number of heaps determines how many recipe options can be used in parallel. The highest mass conversion values are therefore obtained when both parameters are increased together. However, once the available scrap items that can easily satisfy the recipe constraints have been allocated, further increases in capacity mainly provide unused theoretical space rather than additional feasible scrap combinations.

The main finding from Table 6.3 and Figure 6.2 is that mass conversion is primarily limited by available heap capacity at small and medium heap sizes, but increasingly limited by recipe feasibility at larger heap sizes. Increasing heap capacity and the number of heaps improves mass conversion because the algorithm has more space and more recipe options to allocate scrap items. However, the benefit is strongest up to approximately 8 000-10 000 kg. Beyond this range, the curves begin to flatten, especially for configurations with many heaps, indicating that the remaining unallocated scrap becomes harder to combine into feasible recipes under the given chemical constraints. Therefore, simply increasing heap capacity or adding more heaps does not lead to proportional improvements in scrap utilisation. The results show that high mass conversion requires sufficient capacity and multiple available heaps, but also that the system reaches a saturation region where additional capacity produces diminishing returns.

Heap Size (kg)	Number of heaps											
	1	2	3	4	5	6	7	8	9	10	11	12
0	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
1000	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	11%	12%
2000	2%	4%	6%	8%	10%	12%	15%	17%	19%	21%	23%	25%
3000	3%	6%	9%	12%	16%	19%	22%	25%	28%	31%	34%	37%
4000	4%	8%	11%	15%	20%	24%	28%	32%	36%	40%	45%	49%
5000	5%	9%	14%	19%	24%	30%	35%	40%	45%	50%	54%	59%
6000	6%	12%	19%	25%	31%	37%	44%	48%	52%	56%	62%	68%
7000	7%	15%	22%	29%	36%	43%	47%	51%	55%	62%	69%	77%
8000	8%	17%	25%	33%	42%	48%	51%	55%	60%	68%	76%	80%
9000	9%	18%	27%	37%	42%	48%	57%	66%	76%	80%	84%	87%
10000	9%	19%	29%	40%	46%	52%	62%	72%	76%	80%	82%	83%
11000	9%	20%	31%	43%	49%	61%	72%	76%	79%	82%	84%	85%
12000	9%	21%	34%	46%	53%	65%	78%	81%	83%	84%	85%	86%
13000	9%	22%	36%	49%	54%	68%	80%	83%	84%	86%	88%	89%
14000	9%	23%	38%	52%	57%	72%	82%	86%	87%	88%	89%	89%
15000	9%	24%	36%	46%	51%	67%	79%	83%	84%	86%	87%	88%
16000	17%	33%	42%	49%	65%	67%	72%	81%	85%	86%	87%	89%
17000	18%	27%	41%	47%	65%	71%	82%	85%	87%	87%	88%	89%
18000	19%	37%	44%	50%	69%	74%	82%	85%	87%	88%	89%	90%
19000	20%	39%	45%	65%	71%	75%	78%	80%	81%	83%	86%	88%
20000	21%	41%	51%	58%	79%	83%	87%	88%	89%	89%	90%	92%

Table 6.3: Cumulative mass conversion relative to total scrap weight for different numbers of heaps and heap capacities. Results from low to high are shaded from light to dark.

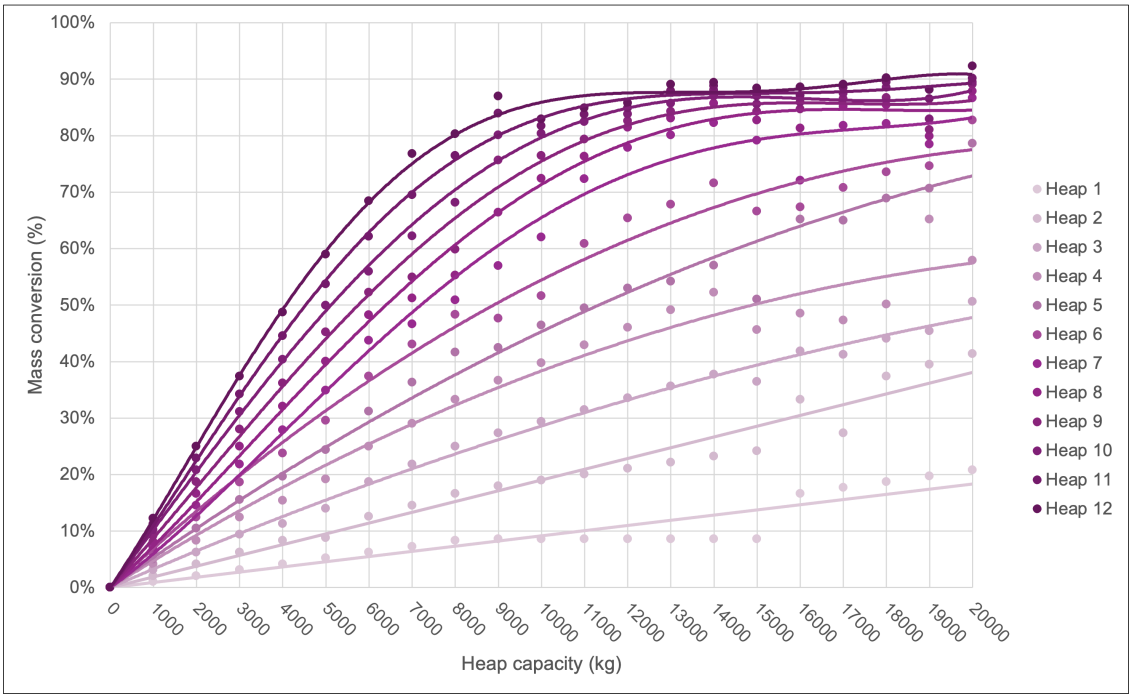


Figure 6.2: Scatter plot of cumulative mass conversion by heap capacity and number of heaps, based on Table 6.3, including polynomial trend lines.

6.4. Heap utilisation performance

Table 6.4 and Figure 6.3 present the cumulative heap utilisation achieved for different heap capacities and numbers of heaps under the baseline configuration. Heap utilisation represents the fraction of available heap capacity that is effectively filled with scrap while satisfying the chemical composition constraints of the selected steel recipes.

For small heap capacities, utilisation is generally high. Up to 3 000 kg, most configurations achieve utilisation values between 99% and 100%, with only minor deviations. At 1 000 kg, utilisation remains between 96% and 100% across all heap counts, while at 2 000 kg all configurations reach 100%. This indicates that, at small capacities, the available scrap provides sufficient flexibility to almost fully utilise the allocated heap space.

As heap capacity increases, heap utilisation decreases most clearly for configurations with larger numbers of heaps. For twelve heaps, utilisation decreases from 98% at 1 000 kg to 80% at 8 000 kg, 66% at 10 000 kg, and 37% at 20 000 kg. This decline of 61 percentage points shows that larger heaps become more difficult to fill completely under the chemical recipe constraints. Larger heaps require more scrap items with compatible compositions, which reduces the probability that the remaining available material can fully satisfy both capacity and composition requirements.

These trends show that heap utilisation is not only determined by the available capacity, but also by the interaction between heap capacity, number of heaps, and recipe feasibility. A larger number of heaps creates more allocation opportunities, but it also divides the available compatible scrap over more recipe-specific outputs. As a result, some heaps remain partly empty even though the total allocated mass may still increase.

The effect becomes more pronounced when the number of heaps increases. At 10 000 kg, utilisation decreases from 83% for one heap to 66% for twelve heaps, a difference of 17 percentage points. At 20 000 kg, this difference increases to 63 percentage points, from 100% for one heap to 37% for twelve heaps. The scatter points and fitted trendlines in Figure 6.3 therefore show that higher heap counts experience a stronger utilisation loss at larger capacities, because multiple heaps compete for suitable scrap combinations.

The main finding from Table 6.4 and Figure 6.3 is that heap utilisation behaves opposite to mass conversion. Small heap capacities are easy to fill almost completely, resulting in utilisation values close to 100%, but they convert only a limited fraction of the total scrap mass. Larger heap capacities allow more scrap to be allocated, but the available capacity becomes increasingly difficult to fill efficiently because each heap must still satisfy the chemical limits of a selected recipe. This effect is strongest when many heaps are used, since multiple heaps compete for compatible scrap items from the same buffer. Therefore, the results show that maximising heap capacity or increasing the number of heaps does not automatically improve space efficiency. Instead, there is a clear trade-off: larger configurations improve mass conversion but reduce heap utilisation, indicating that an optimal configuration must balance total scrap allocation against efficient use of available heap space.

Heap Size (kg)	Number of heaps											
	1	2	3	4	5	6	7	8	9	10	11	12
0	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
1000	100%	100%	100%	100%	96%	97%	97%	98%	98%	98%	98%	98%
2000	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
3000	100%	100%	99%	100%	100%	100%	100%	100%	100%	100%	100%	100%
4000	100%	100%	90%	93%	94%	95%	96%	96%	97%	97%	97%	98%
5000	100%	85%	90%	92%	94%	95%	96%	96%	97%	96%	94%	94%
6000	100%	100%	100%	100%	100%	100%	100%	97%	93%	90%	91%	91%
7000	100%	100%	100%	100%	100%	99%	92%	88%	84%	85%	87%	88%
8000	100%	100%	100%	100%	100%	97%	87%	83%	80%	82%	84%	80%
9000	92%	96%	97%	98%	91%	85%	87%	89%	90%	86%	82%	77%
10000	83%	91%	94%	96%	89%	83%	85%	87%	82%	77%	71%	66%
11000	75%	88%	92%	94%	87%	89%	90%	83%	77%	72%	67%	62%
12000	69%	84%	90%	92%	85%	87%	89%	82%	74%	67%	62%	57%
13000	64%	82%	88%	91%	80%	84%	85%	77%	69%	63%	59%	55%
14000	59%	80%	86%	90%	78%	82%	81%	74%	66%	61%	55%	51%
15000	55%	78%	78%	73%	65%	71%	73%	66%	60%	55%	51%	47%
16000	100%	100%	84%	73%	78%	68%	62%	61%	57%	52%	48%	44%
17000	100%	77%	78%	67%	74%	67%	66%	60%	54%	50%	46%	42%
18000	100%	100%	78%	67%	74%	66%	63%	57%	51%	47%	43%	40%
19000	100%	100%	77%	83%	72%	63%	57%	51%	46%	42%	40%	37%
20000	100%	99%	81%	70%	76%	66%	60%	53%	48%	43%	39%	37%

Table 6.4: Cumulative heap utilisation relative to available heap capacity for different numbers of heaps and heap capacities. Results from low to high are shaded from light to dark.

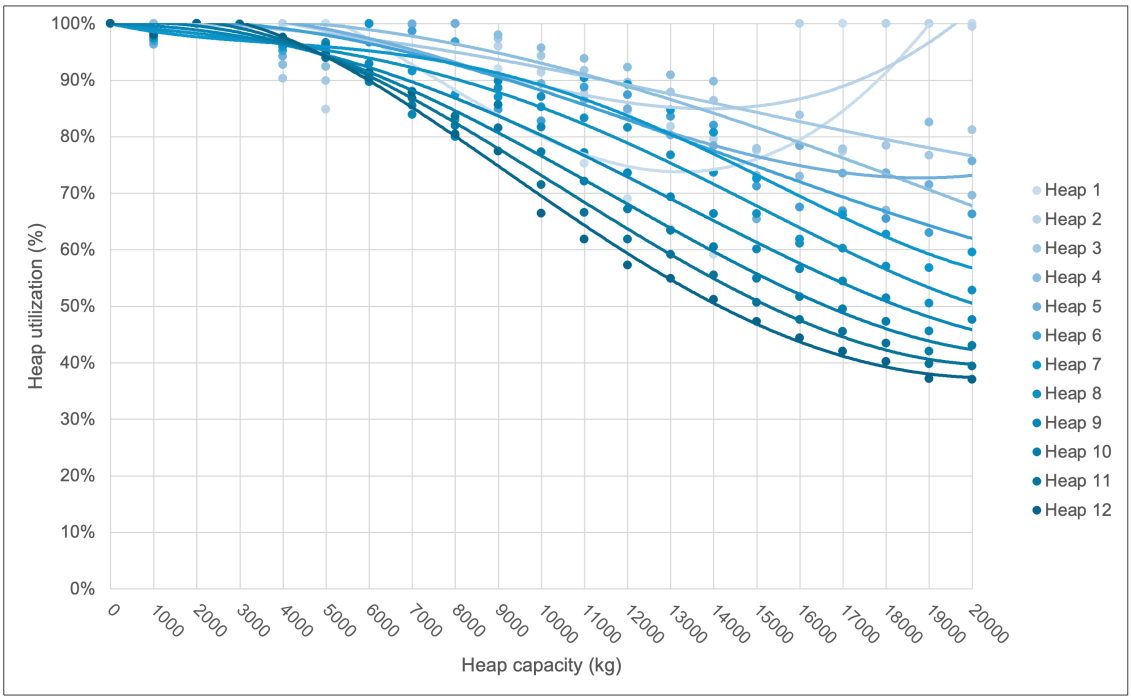


Figure 6.3: Scatter plot of cumulative heap utilisation by heap capacity and number of heaps, based on Table 6.4, including polynomial trend lines.

6.5. Economic performance

Table 6.5 and Figure 6.4 present the cumulative economic value generated for different heap capacities and numbers of heaps under the baseline configuration. The results show how the available heap capacity and the number of simultaneously formed heaps influence the economic performance of the scrap allocation process.

At small heap capacities, cumulative value increases rapidly as capacity grows. For example, with twelve heaps, the total value increases from €60 178 at a heap capacity of 1 000 kg to €146 794 at 3 000 kg and €211 388 at 6 000 kg. This corresponds to an increase of €86 616 between 1 000 kg and 3 000 kg, followed by an additional €64 594 between 3 000 kg and 6 000 kg. This steep increase occurs because larger heap capacities allow more scrap items to be combined into feasible heaps, thereby enabling the production of additional steel recipes.

A similar effect is observed when increasing the number of heaps. For a given heap capacity, forming more heaps allows more steel recipes to be produced simultaneously, which increases the total value generated. For instance, at a heap capacity of 8 000 kg, the cumulative value rises from €54 375 with one heap to €245 616 with twelve heaps. This is an increase of €191 241, or approximately 4.5 times the single-heap value, indicating that economic performance strongly depends on the number of parallel heaps that can be formed.

However, the rate of value growth decreases at larger heap capacities. For twelve heaps, increasing heap capacity from 1 000 kg to 8 000 kg raises value by €185 438, whereas increasing capacity from 8 000 kg to 20 000 kg raises value by only €8 840. The highest value for twelve heaps is reached at 14 000 kg (€270 229), after which the results fluctuate between approximately €233 000 and €257 000. The scatter points and fitted trendlines in Figure 6.4 therefore indicate a clear reduction in marginal economic gains beyond roughly 8 000-10 000 kg.

These diminishing returns occur because most profitable scrap combinations are already captured at moderate heap capacities. Increasing capacity further expands the theoretical search space, but does not necessarily create higher-value feasible recipe combinations. This also explains why the highest heap capacities do not always produce the highest value, despite offering more available capacity. Because the algorithm selects heaps sequentially based on economic value, a change in heap capacity can also change which recipes are selected early in the allocation process, thereby affecting the remaining scrap pool and the value of later heaps.

The main finding from Table 6.5 and Figure 6.4 is that economic performance improves strongly when heap capacity and the number of heaps increase from low to intermediate levels, but the marginal economic benefit decreases at larger capacities. For value creation alone, the best-performing configuration in Table 6.5 is obtained with twelve heaps and a heap capacity of 14 000 kg, producing a cumulative value of €270 229. However, the increase in value becomes much smaller beyond approximately 8 000-10 000 kg, and the results become more variable. This indicates that the system approaches an economic saturation region where additional heap capacity provides limited extra value. Therefore, maximising economic value alone favours relatively large capacities and many heaps, but this does not yet account for how efficiently the available heap space is used. The value results must therefore be interpreted together with the heap utilisation results in the following trade-off analysis.

Heap Size (kg)	Number of heaps											
	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	0	0	0	0	0	0	0	0	0
1000	6800	13599	20398	27196	32753	36753	40753	44752	48750	52748	56578	60178
2000	13600	27199	35199	43195	50395	57595	64795	71995	79195	86395	93595	100794
3000	20399	32399	44210	64602	75402	86202	97002	107800	118600	129400	140194	146794
4000	27200	43198	62509	76909	91309	105708	120108	134508	148906	163306	172105	180904
5000	33998	57682	77681	95681	113681	131681	149680	167680	178680	188681	196732	201982
6000	40798	64797	86397	107997	129596	151192	172778	182490	190802	198788	205088	211388
7000	47598	75441	100641	125841	151039	174412	186723	196433	204271	211621	218970	226320
8000	54375	83175	111975	140774	169573	192877	201788	211045	220766	229166	237566	245616
9000	56290	88690	121089	153486	173543	191362	200812	210262	219691	229151	237228	243613
10000	56290	92289	128289	164287	187326	205287	215787	226287	234800	243142	247771	252337
11000	56290	95890	135489	175088	197790	209340	220890	229128	235801	242306	247241	251255
12000	56290	99490	142689	185885	209835	222434	235034	242605	247025	251177	254834	258218
13000	56290	103089	149889	196687	214295	227945	240439	246572	251551	256367	261138	264842
14000	56290	106690	157088	207484	224008	238708	249497	256834	261324	265442	268013	270229
15000	56290	110290	152728	184466	203190	218940	231627	239213	244560	249511	254185	257310
16000	57600	115200	144874	168142	184942	199222	208975	218360	225585	230266	234545	237650
17000	61200	124439	172705	193580	211430	223804	234914	242039	247191	250852	254246	256656
18000	64800	129596	152548	173569	192469	208804	217486	224459	229370	233181	236470	239040
19000	68400	136800	157425	177375	196148	204648	212835	217519	222070	226067	229680	233121
20000	72000	143207	175357	200369	221369	229919	238351	242919	247067	249652	252232	254456

Table 6.5: Cumulative value creation expressed in euros for different numbers of heaps and heap capacities. Results from low to high are shaded from light to dark.

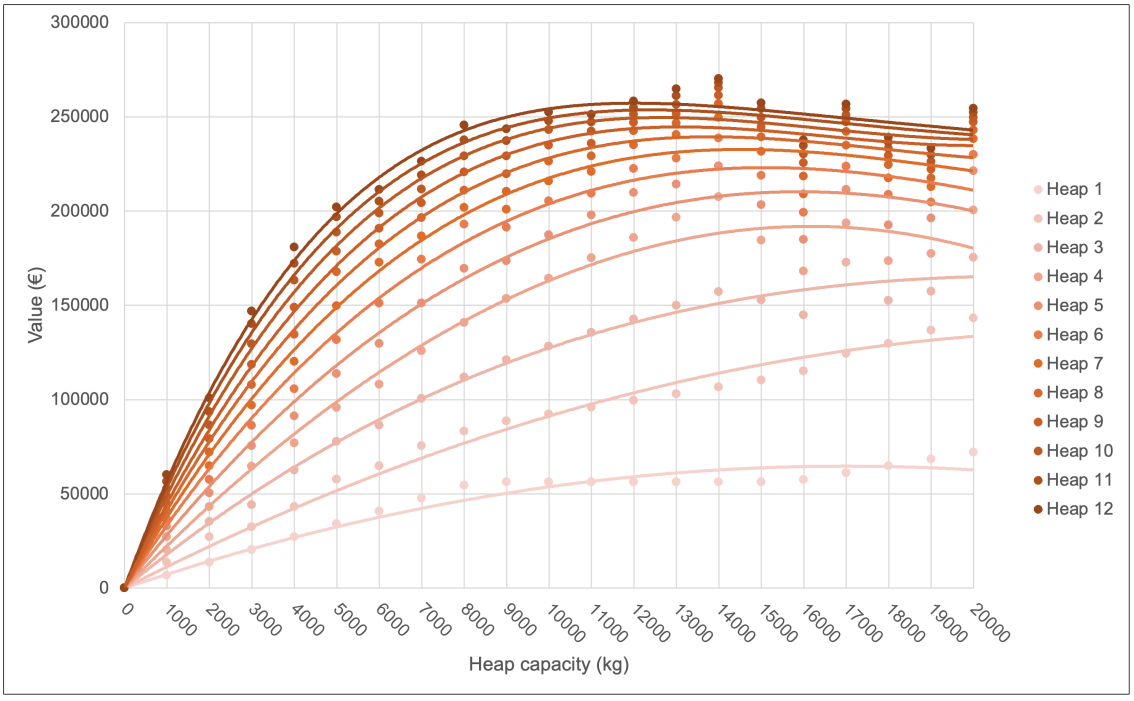


Figure 6.4: Scatter plot of cumulative value creation by heap capacity and number of heaps, based on Table 6.5, including polynomial trend lines.

6.6. Performance trade-offs between heap utilisation and revenue

Figure 6.5 overlays the cumulative value results from Figure 6.4 with the heap utilisation results from Figure 6.3. This visual comparison illustrates the opposing trends that define the system trade-off: economic value generally increases with larger heap capacities and more heaps, whereas heap utilisation tends to decrease as the available heap capacity becomes more difficult to fill efficiently. Table 6.6 and Figure 6.6 quantify this trade-off across the different system configurations. The combined performance metric shown in Table 6.6 is calculated as the cumulative value from Table 6.5 multiplied configuration-wise by the corresponding heap utilisation values from Table 6.4. This metric represents a utilisation-weighted value score, capturing the balance between maximising revenue and efficiently using available heap capacity.

As shown in Table 6.6, performance increases rapidly when heap capacity grows from small values to intermediate levels. For example, with seven heaps, the combined performance increases from 39 685 at 1 000 kg to 172 755 at 6 000 kg and reaches 209 653 at 12 000 kg. Larger heap capacities allow more scrap combinations to satisfy the chemical constraints of the steel recipes, increasing the achievable value. However, this effect is counterbalanced by declining heap utilisation, as larger heaps become more difficult to fill completely with compositionally compatible scrap.

The resulting trade-off is visible in Figure 6.6. For most heap counts, the observed points and fitted trendlines rise sharply at lower capacities, reach a maximum at intermediate capacities, and then decline as utilisation losses outweigh the additional value obtained from larger heaps. For seven heaps, the score decreases from 209 653 at 12 000 kg to 201 485 at 14 000 kg and 141 950 at 20 000 kg. This behaviour reflects the competing effects observed in the previous sections: increasing heap capacity improves mass conversion and economic value, but simultaneously reduces heap utilisation.

The highest combined performance is obtained at a heap capacity of 12 000 kg with seven heaps. In Table 6.6, this configuration yields the maximum performance value of 209 653. Compared with smaller heap capacities, this setup provides sufficient combinatorial flexibility to generate high-value recipes. Compared with larger heap capacities, it avoids the stronger utilisation losses observed beyond the optimum. For example, at 12 000 kg, increasing from seven to twelve heaps reduces the combined score from 209 653 to 147 920, even though the cumulative value itself increases, because heap utilisation decreases from 89% to 57%.

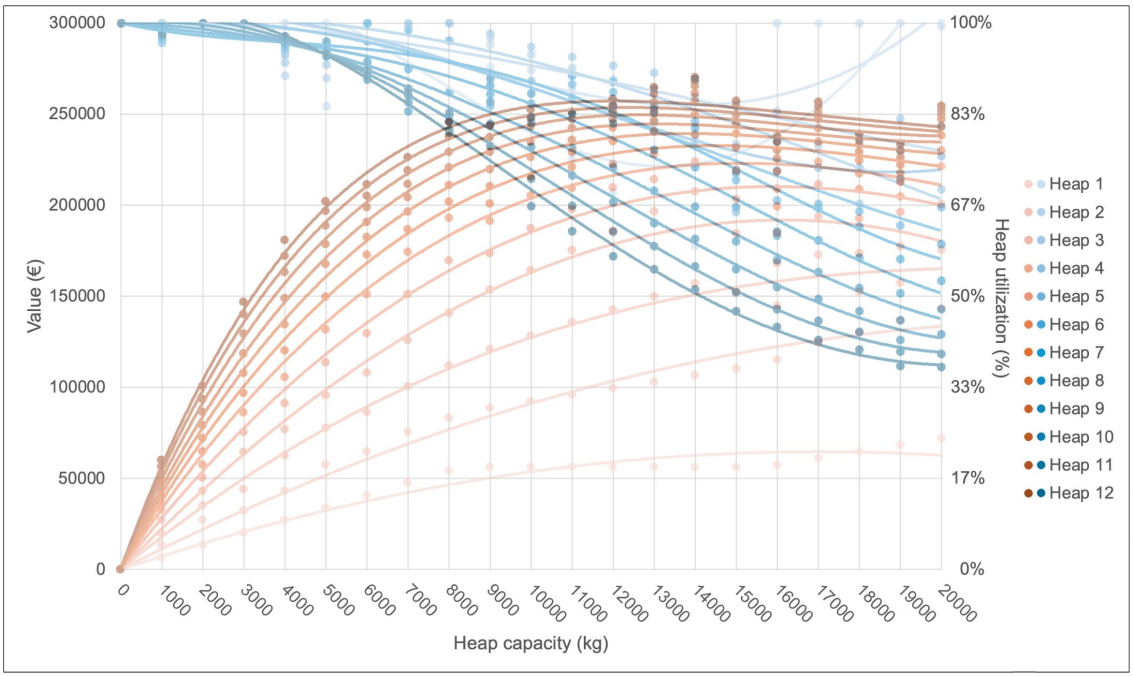


Figure 6.5: Overlay of the cumulative value results from Figure 6.4 and the heap utilisation results from Figure 6.3. The figure illustrates the opposing trends between increasing economic value and decreasing heap utilisation, which together form the basis for the combined performance metric shown in Table 6.6 and Figure 6.6.

Heap Size (kg)	Number of heaps											
	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	0	0	0	0	0	0	0	0	0
1000	6800	13598	20396	27192	31552	35630	39685	43726	47752	51774	55409	59038
2000	13599	27198	35197	43190	50390	57590	64790	71990	79190	86390	93590	100788
3000	20399	32397	43976	64340	75157	85968	96776	107579	118384	129187	139979	146587
4000	27199	43195	56463	71330	86010	100596	115129	129629	144103	158565	167560	176524
5000	33995	48929	69822	88421	106780	125020	143190	161318	172653	181238	184883	190830
6000	40797	64794	86394	107994	129592	151184	172755	176440	177328	178303	185874	193234
7000	47597	75228	100452	125662	150867	172138	170992	172882	171355	180930	190101	198966
8000	54351	83156	111958	140756	169556	186725	176365	175274	176523	187832	198612	197593
9000	51774	85132	117850	150402	157532	162297	174669	186310	197391	196249	193495	188690
10000	46596	84342	120924	157210	167382	169930	183931	197056	191846	188015	177073	167708
11000	42360	84025	124311	164254	171119	185816	199613	190926	181879	174719	164592	155448
12000	38830	84060	127936	171465	178112	194411	209653	198052	181781	168767	157619	147920
13000	35843	84366	131740	178822	171991	190446	203626	189327	174365	162571	154502	145383
14000	33283	84886	135685	186277	175579	195702	201485	189131	173384	160674	148724	138349
15000	31064	85577	119014	134913	132977	155894	168020	158707	146916	136960	128842	121588
16000	57600	115200	121461	122708	144963	134487	129190	133366	127615	118921	111703	105497
17000	61200	96256	134463	129545	155478	149488	155382	145847	134482	124177	115697	107819
18000	64800	129592	119706	116228	141601	136788	136394	128111	118107	110306	102784	96032
19000	68399	136799	120774	146403	140285	128905	120865	109947	101253	94929	91460	86694
20000	72000	142419	142361	139402	167483	152405	141950	128320	117592	107415	99398	94164

Table 6.6: Utilisation-weighted value score calculated by multiplying the cumulative value from Table 6.5 element-wise by the cumulative heap utilisation from Table 6.4. Results from low to high are shaded from light to dark.

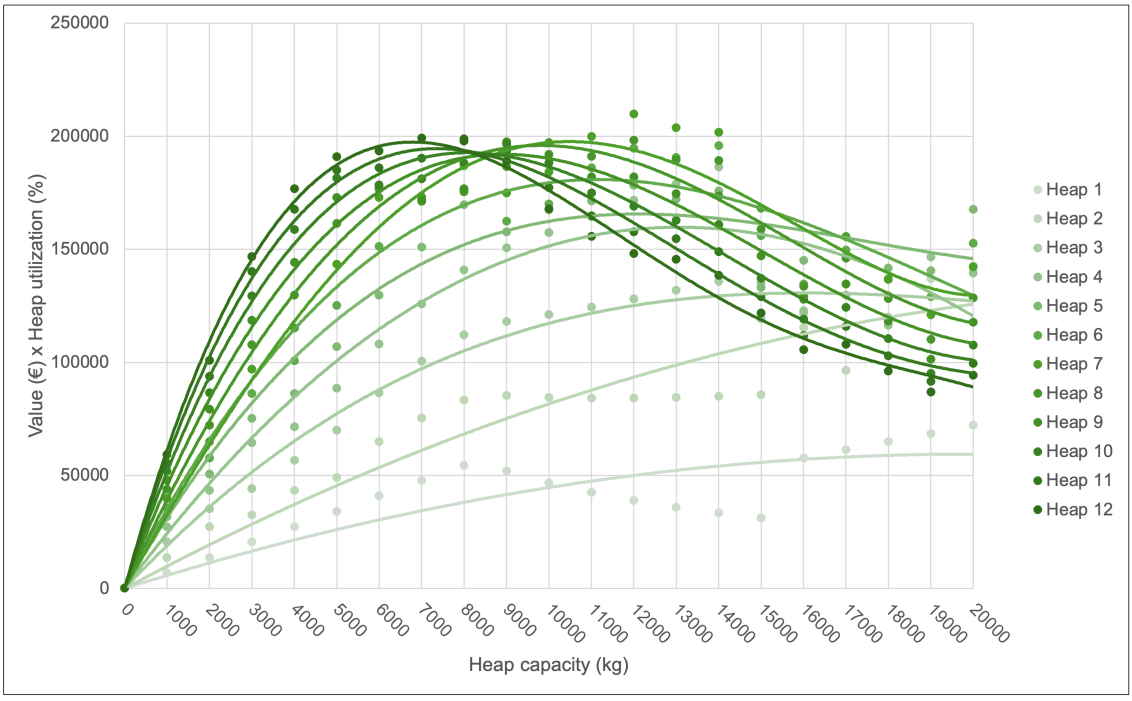


Figure 6.6: Scatter plot of the utilisation-weighted value score by heap capacity and number of heaps, based on Table 6.6, including polynomial trend lines.

The main finding from Figures 6.5 and 6.6 and Table 6.6 is that the best system performance is obtained by balancing heap capacity and number of heaps, rather than maximising either parameter individually. The value results alone favour larger configurations, because more heap capacity and more heaps allow more scrap to be converted into valuable recipes. However, the heap utilisation results show the opposite trend: when the available heap capacity becomes too large, a decreasing fraction of that capacity can be filled with chemically compatible scrap. The combined value-utilisation score therefore identifies the point where additional value creation no longer compensates for the loss in space efficiency.

The highest combined performance is obtained with seven heaps of 12 000 kg each. This configuration reaches a value-utilisation score of 209 653, which is the maximum observed in Table 6.6. Compared with smaller capacities, this configuration provides enough combinatorial flexibility to construct valuable recipe-compliant heaps. Compared with larger capacities or more heaps, it avoids the stronger utilisation losses that reduce the combined score. For example, increasing from seven to twelve heaps at 12 000 kg increases the available sorting capacity, but reduces the combined score from 209 653 to 147 920 because heap utilisation decreases from 89% to 57%. Therefore, the selected configuration represents the most favourable compromise between economic value creation and efficient use of heap capacity.

To further analyse this optimal configuration, Figures 6.7 and 6.8 show the distribution of scrap allocated to each recipe in terms of item conversion and mass conversion. The results indicate that the scrap stream is distributed across several steel grades rather than concentrated in a single recipe. The largest share of scrap items is allocated to S355M, representing 17% of the items and 12% of the mass, followed by X6CrNiNb18-10 with 15% of the items and 13% of the mass. In addition, 18% of the scrap items are recorded as “other scrap” and remain unassigned. This distribution shows that the recipe-based allocation strategy uses multiple steel grades simultaneously to increase conversion and value.

The detailed elemental compositions of the two largest heaps are presented in Figures 6.9 and 6.10, while the remaining heap compositions are provided in Appendix C. In these plots, each point represents an individual scrap item, the black bars indicate the allowable composition range of the corresponding steel recipe, and the red crosses represent the mass-weighted average composition of the allocated scrap items. For all selected heaps, the weighted-average compositions fall within the allowable recipe ranges. This confirms that the optimal configuration not only performs well according to the value-utilisation metric, but also produces chemically feasible heaps.

The composition plots also illustrate the key principle of the proposed sorting strategy. Individual scrap items may lie outside the allowable range for certain elements, but they can still be used when combined with other scrap items that compensate for these deviations. The feasibility of the heap is therefore determined by the mass-weighted average composition rather than by the composition of each item separately. This confirms that the Sorting Decision Layer can use complementary scrap combinations to construct recipe-compliant and economically valuable heaps, while leaving only the scrap items that cannot be feasibly assigned under the selected configuration as unallocated material.

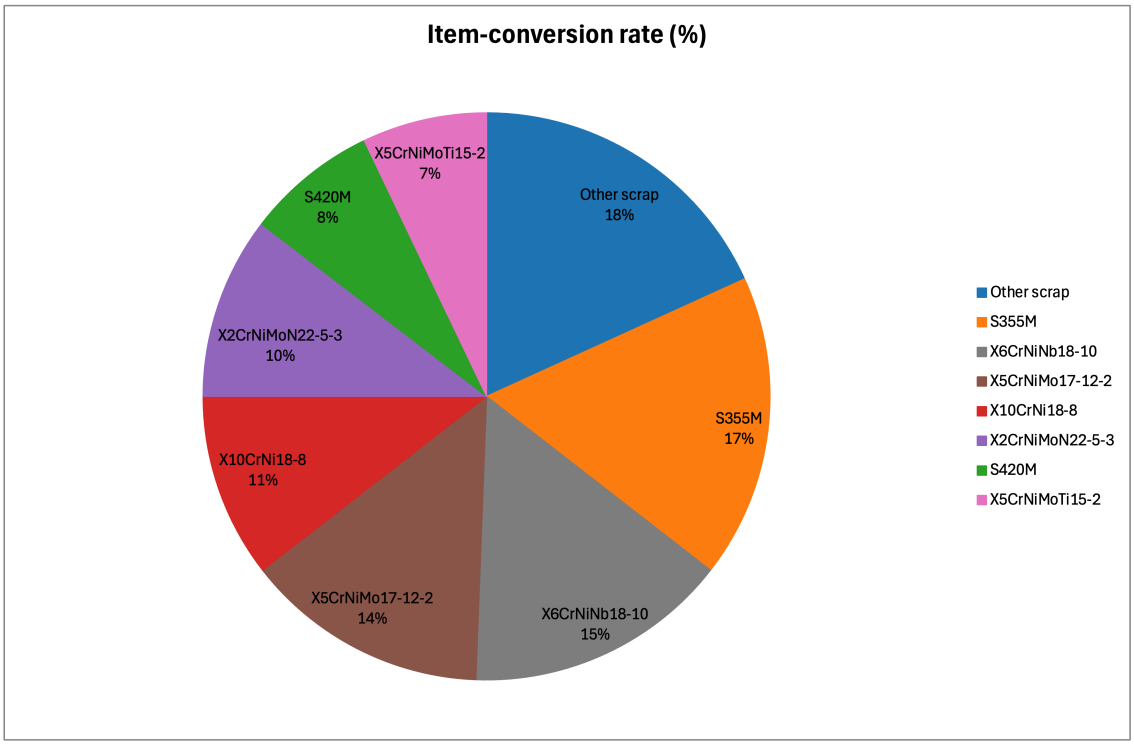


Figure 6.7: Pie chart of the item-conversion rate for the 7-heap configuration with a heap capacity of 12 000 kg per heap.

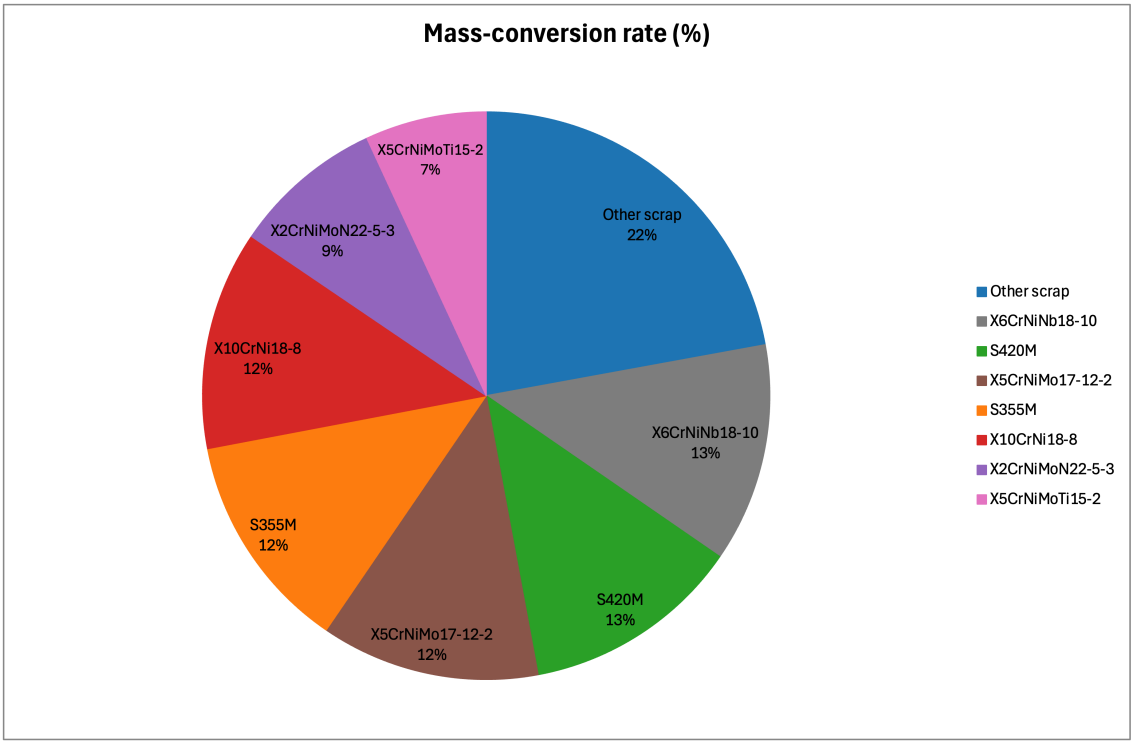


Figure 6.8: Pie chart of the mass-conversion rate for the 7-heap configuration with a heap capacity of 12 000 kg per heap.

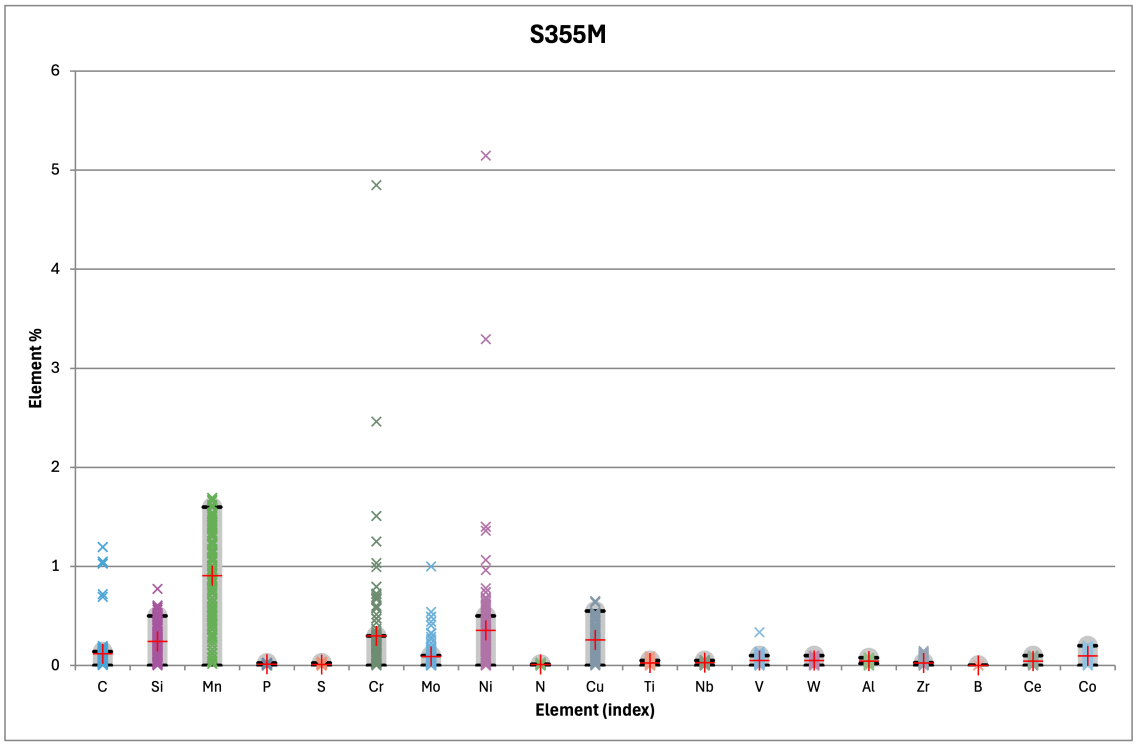


Figure 6.9: Scrap composition per element for the first heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the S355M steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

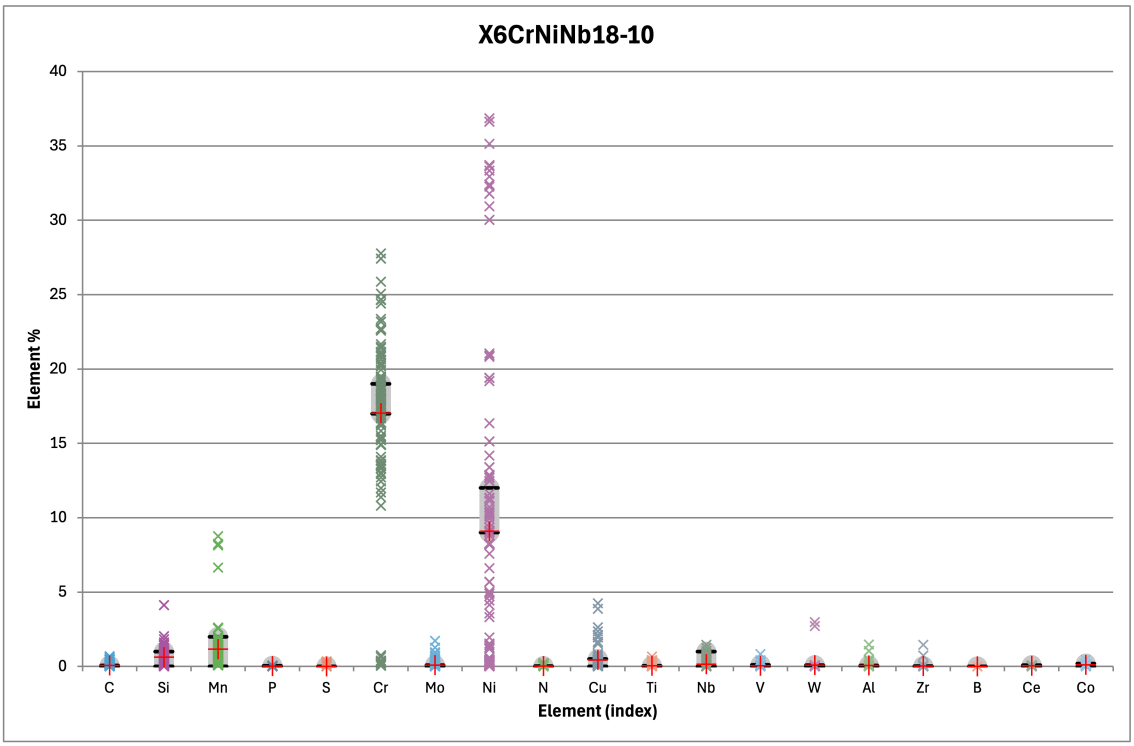


Figure 6.10: Scrap composition per element for the second heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the X6CrNiNb18-10 steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

6.7. Summary of results

This chapter evaluated the performance of the proposed Sorting Decision Layer under different algorithmic and operational configurations. Together, the results answer the fifth sub-research question: "How does the proposed scrap sorting system perform under varying system configurations?" by showing how algorithm choice, buffer size, heap capacity, and number of heaps influence mass conversion, heap utilisation, and economic value creation.

The comparison of candidate algorithms showed that the Multi-pass Greedy algorithm provides the most suitable balance between solution quality and computational efficiency. Although the Multi-start Greedy algorithm achieved the highest conversion rates, its limited value improvement did not justify the substantially higher computation time. The Multi-pass Greedy algorithm was therefore used as the baseline algorithm for evaluating the remaining system configurations.

The buffer-size experiment showed that increasing the number of available scrap items improves the ability of the algorithm to form feasible and valuable heaps, especially at small buffer sizes. However, the improvement rate decreases once the buffer reaches approximately 600-700 scrap items. A buffer size of 1 000 scrap items was therefore selected as the baseline input size, because it captures most of the attainable conversion performance while keeping the computational effort manageable.

The mass conversion, heap utilisation, and economic performance results showed that heap capacity and number of heaps strongly influence system behaviour, but in different directions. Larger capacities and more heaps generally improve mass conversion and value creation by increasing allocation flexibility. However, they also reduce heap utilisation, because the available capacity becomes harder to fill with chemically compatible scrap. Beyond approximately 8 000-10 000 kg, additional capacity produces diminishing returns, indicating that the remaining unallocated scrap becomes increasingly difficult to combine into recipe-compliant heaps.

The combined value-utilisation analysis showed that the most balanced configuration is obtained with seven heaps of 12 000 kg each. This configuration provides the highest combined performance score because it offers enough capacity and recipe flexibility to generate high-value heaps while avoiding the stronger utilisation losses observed at larger capacities or higher heap counts. The best-performing system configuration is therefore not obtained by simply maximising heap capacity or the number of heaps, but by balancing value creation with efficient use of sorting capacity.

The detailed analysis of this configuration further confirmed the feasibility of the recipe-based sorting strategy. The allocation results in Figures 6.7 and 6.8 show that scrap is distributed across multiple steel recipes, while the composition plots in Appendix C show that the mass-weighted average compositions of the allocated heaps remain within the allowable recipe limits. This confirms the central principle of the Sorting Decision Layer: **instead of discarding scrap items with extreme compositions, the algorithm combines complementary scrap pieces to construct feasible and economically valuable steel recipes.**

Overall, the proposed scrap sorting system performs best under intermediate configurations that provide sufficient allocation flexibility without excessive unused capacity. The results show that recipe-based scrap allocation can improve utilisation and value recovery, while also revealing limitations caused by recipe feasibility and the sequential behaviour of the greedy algorithm. These findings form the basis for the next chapter (Chapter 7), where the implications, limitations, and possible improvements of the proposed approach are evaluated.

7

Discussion

The results showed that the proposed Sorting Decision Layer can construct recipe-compliant scrap heaps and that its performance depends strongly on the balance between heap capacity, number of heaps, buffer size, and algorithmic efficiency. However, these results also need to be interpreted in relation to the assumptions and practical constraints of the simulated system. This chapter therefore discusses what the observed performance means for recipe-based scrap sorting, to what extent the proposed approach could be implemented in practice, and which limitations must be considered before translating the simulation results to an industrial setting.

This chapter discusses the implications of the simulation results presented in Chapter 6. First, Section 7.1 interprets the main trends and trade-offs observed in the experiments. Next, Section 7.2 evaluates the practical feasibility and scalability of the proposed recipe-based scrap sorting system. The main methodological and practical limitations are then addressed in Section 7.3. Section 7.4 compares the proposed recipe-based allocation approach with existing scrap sorting methods, and the chapter is concluded in Section 7.5

7.1. Interpretation of simulation results

The simulation results demonstrate that a carefully tuned, recipe-based sorting decision layer can achieve a favourable balance between material utilisation, economic performance, and computational feasibility. The algorithm comparison shows that the multi-pass greedy algorithm provides the best trade-off between mass conversion and computation time (Table 6.1). While multi-start variants achieve slightly higher conversion, their computation times increase by an order of magnitude, which limits their suitability for iterative simulations. The multi-pass greedy approach is therefore a valid choice, delivering near-best observed mass conversion at acceptable computational cost.

The buffer size analysis indicates that system performance is sensitive to buffer size at small scales, but stabilises beyond 1 000 scrap items (Figure 6.1). At smaller buffer sizes, limited combinatorial flexibility restricts feasible recipe formation, leading to lower conversion and value. Once the buffer exceeds approximately 700-1 000 items, additional scrap availability yields diminishing returns, with mass conversion fluctuating around a stable plateau. This suggests that, beyond a certain threshold, the system captures most relevant combinatorial opportunities. A further increase in buffer size adds computational burden without substantial performance gains. The selected buffer size of 1 000 scrap pieces is therefore a stable operating point.

Under the baseline configuration, the results reveal a clear trade-off between mass conversion, heap utilisation, and economic value. Increasing heap capacity and the number of heaps improves mass conversion, but reduces average heap utilisation due to the reduced ability to fully fill large heaps within compositional constraints. The combined performance metric shown in Table 6.6 highlights an optimal region around seven heaps with a heap capacity of 12 000 kg, where approximately 78 % mass conversion (Table 6.3) is achieved with 89 % heap utilisation (Table 6.4), while maintaining strong economic performance (Table 6.5). This operating region reflects a balanced system configuration in which sufficient flexibility exists to form valuable, recipe-compliant heaps without excessive underutilisation of capacity. Overall, the results demonstrate that system-level parameter tuning is at least as important as algorithm selection in maximising the effectiveness of recipe-based scrap sorting.

7.2. Practical feasibility and scalability

The proposed recipe-based scrap sorting system is practically feasible, provided that it is implemented as a decision-support layer on top of existing sensor-based sorting infrastructure. The simulation assumes that individual scrap items can be identified by composition and mass with sufficient accuracy, which aligns with the capabilities of modern sensor technologies discussed in Section 2.3. The allocation logic itself does not require real-time optimisation at millisecond scale. Instead, it can operate in batches, making integration with industrial material handling systems more realistic.

From a computational perspective, the selected multi-pass greedy algorithm demonstrates acceptable runtimes for the tested problem sizes, even with multiple heaps and large buffer sizes. This suggests that the approach can scale to industrial scrap volumes. In addition, the algorithmic structure allows for gradual simplification if computational limits are reached. For example, the number of passes, evaluated heaps, or candidate recipes can be reduced without causing system failure or requiring a fundamentally different model.

Scalability in physical systems is mainly governed by buffer capacity, heap layout, and logistics rather than algorithmic complexity alone. The results show that performance saturates beyond certain buffer sizes and heap capacities, indicating that larger installations do not necessarily require proportionally larger buffers to remain effective. This is favourable for industrial deployment, as it limits space and handling requirements.

The design of the system supports incremental implementation. Steel grades, recipes, and economic parameters can be updated independently, allowing adaptation to changing market conditions or production priorities. Overall, the proposed system is scalable both computationally and operationally, provided that design parameters are tuned to the specific industrial context.

7.3. Limitations of the proposed approach

Despite the promising results, the proposed scrap sorting approach has several limitations that should be mentioned. First, the simulation assumes accurate and complete knowledge of the chemical composition and mass of individual scrap items. In practice, sensor-based measurements are subject to noise, uncertainty, surface contamination, and incomplete elemental coverage. These effects are not explicitly modelled and may reduce achievable performance in real-world applications.

A second limitation concerns the use of synthetically generated scrap data. Although synthetic data enable controlled simulation experiments and make it possible to systematically vary buffer size, heap capacity, and heap count, they may not fully represent the statistical structure of real industrial scrap streams. Real scrap is likely to contain correlations between alloying elements, product-origin clusters, irregular mass distributions, contamination patterns, and temporal variations in material arrivals. These effects may influence the availability of feasible combinations and could make recipe-compliant heap formation more difficult than suggested by the synthetic dataset. The results should therefore be interpreted as a proof-of-concept and parameter sensitivity analysis rather than as a direct prediction of industrial performance. Future work should validate and calibrate the model using measured composition and mass data from actual scrap streams.

The steel composition data are derived from standard limits using simplified assumptions, such as selecting the strongest configurations and excluding summation rules such as carbon equivalent constraints (CEV). While this ensures consistent numerical input, it does not fully capture the complexity of industrial steelmaking recipes. Dynamic effects such as changing order books, furnace availability, alloy correction strategies, and production scheduling are outside the scope of the model and this research.

From an algorithmic perspective, the greedy-based approach does not guarantee global optimality. Although multi-pass strategies significantly improve solution quality, the method may still miss better combinations in highly constrained or uneven scrap pools. More advanced optimisation techniques could yield higher performance, but were computationally infeasible with the available hardware (section 4.2).

Finally, logistical and operational constraints such as conveyor capacities, storage layout, crane movements, pile accessibility, and safety considerations are abstracted into simplified buffers and heaps. These factors may limit practical implementation and require additional modelling before industrial deployment.

7.4. Comparison with existing scrap sorting methods

Industrial steel recycling already achieves high recovery rates. Around 85-90% of steel from manufacturing waste and end-of-life products is currently recovered as steel scrap for new steel production, according to Javier Bonaplata in an article for the World Economic Forum Annual Meeting of 2023 [55]. The optimised system's 78% mass conversion is of the same order of magnitude, although it should not be interpreted as directly equivalent to conventional recovery rates, since the proposed system measures conversion into recipe-compliant heaps rather than general scrap recovery. However, unlike a bulk-sorted scrap approach, which may require dilution of impurities or direct scrap towards lower-value applications [2], the recipe-driven method selectively allocates scrap to meet more valuable composition recipes. This allows mixed scrap to be used more strategically in higher-grade steel applications, maintaining product quality by matching scrap combinations to chemical recipe constraints rather than relying mainly on primary metals to compensate for impurity issues.

The recipe-optimised approach adds economic value by improving the use of scrap as a primary resource. Steel scrap has high intrinsic value as a substitute for iron ore in making new steel [55]. By intelligently selecting and blending scrap to limit contaminant build-up, the system can reduce dependence on virgin alloying additions. Raabe et al. [2] show that insufficiently sorted scrap can be unsuitable for high-performance steels because these grades require narrow chemical tolerances and strict quality control. By targeting high-grade, high-value recipes from mixed scrap, the optimised system not only yields competitive material conversion but also improves value capture. It allows recyclers and steelmakers to obtain more economic value per tonne of scrap through recipe-compliant outputs, thereby improving the competitiveness of renewable steel production.

7.5. Chapter summary

This chapter discussed the implications, feasibility, limitations, and added value of the proposed recipe-based scrap sorting approach. The results show that the Sorting Decision Layer can construct chemically feasible heaps and that its performance depends strongly on the balance between buffer size, heap capacity, number of heaps, and algorithmic efficiency. The most favourable results are obtained under intermediate configurations, where sufficient allocation flexibility is available without excessive unused heap capacity.

The discussion also showed that the proposed system is most realistic as a decision-support layer built on top of existing sensor-based sorting infrastructure. However, the results should be interpreted as a proof of concept, because the simulation relies on synthetic scrap data, simplified recipe constraints, and idealised composition measurements. Further validation with industrial scrap data and real sensor uncertainty is therefore required before practical implementation.

Overall, the proposed approach demonstrates how item-level composition data can be translated into recipe-compliant and economically valuable scrap allocations. The Conclusion (Chapter 8) brings these findings together by answering the main research question, summarising the main contributions of the thesis, and outlining directions for future work.

8

Conclusion

The previous chapters developed the thesis from the need for higher-value steel scrap recycling towards the design, simulation, and evaluation of a recipe-based Sorting Decision Layer. The literature review showed that sensor-based sorting technologies can provide piece-level information, while the system design translated this information into a recipe-based allocation model. The candidate algorithms and simulation experiments then demonstrated how this allocation problem can be evaluated under realistic operational constraints. The results and discussion showed that complementary scrap pieces can be combined into chemically feasible and economically valuable heaps, but also that performance depends on algorithm choice, buffer size, heap capacity, and the number of available heaps. This chapter brings these findings together to answer how sensor-based scrap sorting can be developed and simulated to enable recipe-based steel production from large volumes of scrap metal.

The chapter concludes the thesis by summarising the main findings of the developed and simulated scrap sorting decision layer. First, Section 8.1 summarises the answers to the sub-research questions. Next, Section 8.2 draws conclusions with respect to the main research question. Finally, Section 8.3 provides recommendations for industrial implementation, and Section 8.4 outlines directions for future research.

8.1. Summary answers to sub-research questions

The literature review (Chapter 2) showed that sensor-based sorting technologies such as LIBS, XRF, and XRT can provide information on material type, density, and elemental composition at industrially relevant throughputs. Reported throughputs of approximately 3-11t/h and purities above 95% indicate that the hardware foundation for recipe-based scrap sorting already exists. The main research gap is therefore not sensing capability itself, but the lack of system-level allocation strategies that use piecewise composition data to support recipe-based steel production.

A sorting decision layer was developed (Chapter 3) in which scanned scrap items are temporarily stored in a buffer and then algorithmically assigned to steel recipes. Recipe constraints were derived from European Steel Standards and translated into lower and upper elemental bounds. Potential heaps were evaluated using mass-weighted average compositions, allowing sorted scrap streams to be formed based on recipe compliance rather than broad material categories.

The allocation problem is highly combinatorial, making exact optimisation difficult to apply at the required scale. Greedy heuristic algorithms were found to be more suitable because they provide feasible solutions within realistic computation times (Chapter 4). Among the tested approaches (Section 6.1), the multi-pass greedy algorithm achieved the best balance between solution quality and computational efficiency, while multi-start variants provided only limited performance gains at substantially higher computational cost.

System performance (Section 5.1) was evaluated using item conversion, mass conversion, heap utilisation, economic value, and throughput. These indicators enabled the comparison of different combinations of tuning parameters (Section 5.2.1) in terms of material utilisation, space efficiency, and economic performance. The results show that performance tuning requires balancing value creation against heap utilisation (Section 6.6), since no single configuration maximises all indicators simultaneously.

The simulation results (Chapter 6) show that buffer size, heap capacity, and the number of heaps strongly influence system performance. Small buffers limit feasible combinations, while performance gains diminish beyond approximately 700-1 000 scrap items. Increasing heap capacity and the number of heaps generally improves mass conversion and economic value, but may reduce heap utilisation. The best-balanced operating region was found around seven heaps with a heap capacity of 12 000 kg, achieving approximately 78% mass conversion and 89% heap utilisation while maintaining strong economic performance.

8.2. Conclusions with respect to the main research question

The main research question was:

> *"How can a sensor-based scrap sorting system be developed and simulated to enable recipe-based steel production and maximise the value from large volumes of scrap metal?"*

This research concludes that a scrap sorting decision layer can be developed by combining piecewise scrap composition data, recipe constraints derived from European Steel Standards (Section 5.3.2), and scalable allocation algorithms. The proposed system shifts scrap sorting from a separation-oriented process toward a recipe-based allocation process, in which the value of a scrap item depends on how well it contributes to a chemically compliant and economically valuable steel heap (recipe).

The simulation framework demonstrates that recipe-based steel production from mixed scrap streams is technically feasible under the assumptions (Section 5.3.1) of the model. By calculating mass-weighted elemental compositions for candidate heaps (Equation (3.2)), the system can determine whether selected scrap items satisfy the chemical bounds of specific steel recipes. This makes it possible to form furnace-ready scrap heaps without relying solely on conventional scrap categories or excessive dilution with newly mined material.

The results (Chapter 6) also show that maximising value from large volumes of scrap metal requires both suitable algorithms and appropriate system configuration. Greedy heuristic algorithms, particularly the multi-pass greedy algorithm, provide a practical solution to the combinatorial allocation problem. However, algorithm selection alone is not sufficient. Buffer size, heap capacity, and the number of

available heaps strongly influence the degree to which the system can convert incoming scrap into valuable recipe-compliant heaps. Therefore, the proposed sorting decision layer should be understood as an integrated decision-support framework in which sensing, buffering, recipe selection, and allocation logic work together.

Overall, this thesis demonstrates that recipe-based scrap sorting is a promising strategy for increasing the value of recycled steel scrap. By improving the economic competitiveness of recycled steel relative to primary steelmaking from virgin raw materials, recipe-based scrap sorting can contribute to a more sustainable steel production system. While the system does not guarantee globally optimal solutions, it achieves strong performance within realistic computational limits. The proposed approach therefore provides a viable blueprint for integrating modern sensor-based sorting technologies into value-oriented and more targeted steel recycling strategies.

8.3. Recommendations for industrial implementation

The results of this study show that industrial scrap sorting operations would benefit from a transition from bulk classification toward piece-level, recipe-based allocation strategies. Rather than treating sorting as a purely separation-driven task, the proposed system demonstrates the value of incorporating chemical feasibility, economic value, and logistical constraints directly into the allocation logic.

A buffered sorting design is essential in this context, as it provides the flexibility required to assemble chemically compliant scrap mixtures from mixed input streams. The simulations show that small buffers limit achievable performance, while moderate buffer sizes enable substantial gains in conversion and value. However, the observed diminishing returns at larger buffer sizes indicate that industrial implementation should not simply maximise buffer capacity, but instead determine a balanced buffer size based on available space, handling capacity, and expected scrap variability.

From an algorithmic perspective, greedy heuristic approaches offer a practical and scalable solution for industrial deployment. They achieve high-quality results within realistic computational limits and can be simplified when faster decision-making is required. The integration of economic feedback into sorting decisions is also recommended, as it allows the system to prioritise high-value steel recipes and adapt to changing market conditions.

Although this thesis focuses on steel scrap and steel recipes, the proposed Sorting Decision Layer should be implemented in a modular way so that the recipe library can be extended beyond steel. The underlying allocation principle is based on item mass, elemental composition, recipe limits, and material value. This makes the framework potentially applicable to other metal systems, such as aluminium and copper alloys, provided that suitable composition limits, sensor data, and market values are available.

The proposed framework can be implemented incrementally. In the short term, it could function as a decision-support tool that recommends recipe-compliant scrap allocations to operators. In later stages, it could be integrated with automated sorting lines, buffer management systems, and production planning software. For successful industrial deployment, the system should be calibrated using site-specific scrap data, available steel recipes, market prices, sensor accuracy, and logistical constraints.

8.4. Recommendations for future research

Although this research demonstrates the technical feasibility and potential benefits of recipe-based scrap sorting, several topics remain open for future investigation. These recommendations directly follow from the limitations discussed in Section 7.3.

First, future research should incorporate sensor uncertainty into the simulation model. The current model assumes accurate and complete knowledge of the mass and chemical composition of each scrap item. In practice, sensor measurements may be affected by noise, surface contamination, object geometry, incomplete elemental coverage, and classification errors. Including these uncertainties would make the model more representative of industrial operating conditions and would allow the robustness of recipe-based allocation to be evaluated.

Second, the model should be validated using real industrial scrap data. The present simulation relies on synthetically generated scrap compositions, which are useful for controlled experiments but may not

fully represent real scrap streams. Actual scrap data may contain correlations between elements, recurring product-origin patterns, irregular mass distributions, and temporal variation in material arrivals. Validation with measured per-piece sensor data would therefore be an important step toward determining whether the observed conversion rates, heap utilisation, and value creation can also be achieved in practice.

Third, future research should refine the representation of steel recipes and production constraints. This thesis used chemical bounds derived from European Steel Standards, but simplified several aspects of industrial steelmaking, such as carbon equivalent constraints, summation rules, furnace corrections, order books, and production scheduling. Including these factors would allow the allocation model to better reflect the constraints faced by steelmakers during actual production.

Fourth, future work could explore more advanced optimisation approaches. The greedy-based algorithms used in this thesis were selected because of their computational efficiency, but they do not guarantee global optimality. Hybrid methods that combine greedy allocation with local search, metaheuristics, or selective exact optimisation may improve solution quality while remaining computationally feasible. Such methods should be evaluated not only on conversion and value, but also on runtime and scalability.

Fifth, future research should investigate whether the proposed recipe-based allocation framework can be transferred to other metal streams, such as aluminium and copper alloys. This would require alloy-specific recipe libraries, suitable sensor data, and validation experiments to determine whether the same mass-weighted allocation principle can improve value recovery beyond steel scrap sorting.

Finally, future research should extend the model with logistical and operational constraints. The current simulation abstracts the physical system into buffers and heaps, while real sorting facilities are limited by conveyor capacities, crane movements, pile accessibility, storage layout, safety requirements, and furnace charging procedures. Incorporating these constraints would support the transition from a simulation-based proof of concept toward an implementable industrial decision-support system.

Together, these recommendations indicate that the proposed Sorting Decision Layer should be seen as a proof of concept and a basis for further development rather than as a final industrial solution. Future work should therefore focus on validating the approach with industrial scrap data, incorporating sensor uncertainty, and extending the allocation model towards more realistic steelmaking and logistical constraints. Nevertheless, the results of this thesis show that recipe-based scrap sorting can translate piece-level composition data into chemically feasible and economically valuable scrap allocations. In this way, the developed approach provides a practical foundation for increasing the value recovered from recycled steel scrap and reducing the dependence on primary raw materials in steel production.

References

[1] World Steel Association. *World Steel in Figures 2025*. Statistical Report. Accessed: 26 January 2026. World Steel Association, 2025. url: https://worldsteel.org/wp-content/uploads/World-Steel-in-Figures-2025.pdf.

[2] Dierk Raabe et al. “Circular steel for fast decarbonization: thermodynamics, kinetics, and microstructure behind upcycling scrap into high-performance sheet steel”. In: *Annual review of materials research* 54.1 (2024), pp. 247–297.

[3] EuRIC AISBL. *Metal Recycling Factsheet*. EuRIC AISBL – Recycling: Bridging Circular Economy & Climate Policy. Accessed in 2026. 2019. url: https://www.euric-aisbl.eu/.

[4] Ishwar Kapoor, Claire Davis, and Zushu Li. “Effects of residual elements during the casting process of steel production: a critical review”. In: *Ironmaking & steelmaking* 48.6 (2021), pp. 712–727.

[5] Reinol Josef Compañero et al. “Appraising the value of compositional information and its implications to scrap-based production of steel”. In: *Mineral Economics* 36.3 (2023), pp. 463–480.

[6] Beatriz Pérez Horno, Andreas Feldmann, and Peter Samuelsson. “Mapping the dynamics shaping the future of steel recycling”. In: *Discover Sustainability* 6.1 (2025), p. 808.

[7] Leslie Brooks and Gabrielle Gaustad. “The potential for XRF & LIBS handheld analyzers to perform material characterization in scrap yards”. In: *Journal of Sustainable Metallurgy* 7.2 (2021), pp. 732–754.

[8] Max Kölking et al. “More resource efficient recycling of copper and copper alloys by using X-ray fluorescence sorting systems: An investigation on the metallic fraction of mixed foundry residues”. In: *Waste Management & Research* 42.9 (2024), pp. 814–822.

[9] Dillam Jossue Díaz-Romero et al. “Real-time classification of aluminum metal scrap with laser-induced breakdown spectroscopy using deep and other machine learning approaches”. In: *Spectrochimica Acta Part B: Atomic Spectroscopy* 196 (2022), p. 106519.

[10] Waseem S Khan et al. “Recycling and reusing of engineering materials: Recycling for sustainable developments”. In: (2022).

[11] Takuma Watari et al. “Global stagnation and regional variations in steel recycling”. In: *Resources, Conservation and Recycling* 220 (2025), p. 108363.

[12] Bo Björkman and Caisa Samuelsson. “Recycling of steel”. In: *Handbook of recycling*. Elsevier, 2014, pp. 65–83.

[13] Vaclav Smil. *Still the iron age: iron and steel in the modern world*. Butterworth-Heinemann, 2016.

[14] J Svoboda and T Fujita. “Recent developments in magnetic methods of material separation”. In: *Minerals Engineering* 16.9 (2003), pp. 785–792.

[15] York R Smith, James R Nagel, and Raj K Rajamani. “Eddy current separation for recovery of non-ferrous metallic particles: A comprehensive review”. In: *Minerals Engineering* 133 (2019), pp. 149–159.

[16] Ishwar Kapoor, Claire Davis, and Zushu Li. “Effect of Residual Elements during the Hot-Working Process of Steel Production: A Critical Review”. In: *steel research international* 95.9 (2024), p. 2400116.

[17] Zhe Huang et al. “Eddy current separation can be used in separation of non-ferrous particles from crushed waste printed circuit boards”. In: *Journal of Cleaner Production* 312 (2021), p. 127755.

[18] Kane C Williams, Michael D O’Toole, and Anthony J Peyton. “Scrap metal classification using magnetic induction spectroscopy and machine vision”. In: *IEEE Transactions on Instrumentation and Measurement* 72 (2023), pp. 1–11.

[19] Christopher Robben et al. “X-ray-transmission based ore sorting at the San Rafael tin mine”. In: *Minerals Engineering* 145 (2020), p. 105870.

[20] Mary Cesetti and Piergiorgio Nicolosi. “Waste processing: new near infrared technologies for material identification and selection”. In: *Journal of Instrumentation* 11.09 (2016), p. C09002.

[21] Wenguang Xu et al. “Classification and rating of steel scrap using deep learning”. In: *Engineering applications of artificial intelligence* 123 (2023), p. 106241.

[22] TOMRA Recycling. *AUTOSORT™ PULSE: Dynamic LIBS Sorting System*. Accessed: 2025-12-16. url: https://www.tomra.com/waste-metal-recycling/products/machines/autosort-pulse.

[23] Ocean Optics. *High-Throughput Sensing System for Precision Scrap Metal Sorting*. Accessed: 2025-12-16. url: https://www.oceanoptics.com/metal-recycling/high-throughput-sensing-system-provides-precision-sorting-of-scrap-aluminum/.

[24] STEINERT GmbH. *STEINERT PLASMAX: LIBS Sorting System for Metal Recycling*. Accessed: 2025-12-16. url: https://steinertglobal.com/us/sorting-systems/sensor-sorting/libs-sorting-systems/steinert-plasmax-libs/.

[25] SGM Magnetics. *X-ray Separators: XRT and XRF Sorting Systems*. Accessed: 2025-12-16. url: https://www.sgmmagnetics.com/en/products/x-ray-separators-xrt-xrf-t/.

[26] SGM Magnetics. *LIBS Separator Cleansort R*. https://www.sgmmagnetics.com/en/products/libs-separator-cleansort-r/. Accessed: 2025-12-16. 2025.

[27] Klaus-P Bernatzki et al. “Optimal scrap combination for steel production”. In: *Operations-Research-Spektrum* 20.4 (1998), pp. 251–258.

[28] Yongli Wu, Tijmen Oudshoorn, and Peter Rem. “Modelling and optimization of an innovative facility for automated sorting of aluminium scraps”. In: *Waste Management* 189 (2024), pp. 103–113.

[29] Kane C Williams et al. “Classification of shredded aluminium scrap metal using magnetic induction spectroscopy”. In: *Sensors* 23.18 (2023), p. 7837.

[30] Dillam Diaz-Romero et al. “Classification of aluminum scrap by laser induced breakdown spectroscopy (LIBS) and RGB+ D image fusion using deep learning approaches”. In: *Resources, Conservation and Recycling* 190 (2023), p. 106865.

[31] STEINERT GmbH. *STEINERT LSS® | LIBS*. Accessed: 2026-01-15. url: https://steinertglobal.com/sorting-systems/sensor-sorting/libs-sorting-systems/steinert-lss-libs/.

[32] Hans Kellerer, Ulrich Pferschy, and David Pisinger. “Multidimensional knapsack problems”. In: *Knapsack problems*. Springer, 2004, pp. 235–283.

[33] Jakob Puchinger, Günther R Raidl, and Ulrich Pferschy. “The multidimensional knapsack problem: Structure and algorithms”. In: *INFORMS Journal on Computing* 22.2 (2010), pp. 250–265.

[34] George L Nemhauser and Laurence A Wolsey. *Integer and combinatorial optimization*. Vol. 18. Wiley New York, 1988.

[35] David R Morrison et al. “Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning”. In: *Discrete optimization* 19 (2016), pp. 79–102.

[36] John E Mitchell. “Branch-and-cut algorithms for combinatorial optimization problems”. In: *Handbook of applied optimization* 1.1 (2002), pp. 65–77.

[37] Holger H Hoos and Thomas Stützle. “Stochastic local search algorithms: an overview”. In: *Springer Handbook of Computational Intelligence* (2015), pp. 1085–1105.

[38] *NEN-EN 10025-2:2019 – Hot rolled products of structural steels – Part 2: Technical delivery conditions for non-alloy structural steels*. NEN. Delft, The Netherlands, 2019.

[39] *NEN-EN 10025-3:2019 – Hot rolled products of structural steels – Part 3: Technical delivery conditions for normalized/normalized rolled weldable fine grain structural steels*. NEN. Delft, The Netherlands, 2019.

[40] *NEN-EN 10025-4:2019+A1:2022 – Hot rolled products of structural steels – Part 4: Technical delivery conditions for thermomechanical rolled weldable fine grain structural steels*. NEN. Delft, The Netherlands, 2022.

[41] *NEN-EN 10025-5:2019 – Hot rolled products of structural steels – Part 5: Technical delivery conditions for structural steels with improved atmospheric corrosion resistance*. NEN. Delft, The Netherlands, 2019.

[42] *NEN-EN 10025-6:2019+A1:2022 – Hot rolled products of structural steels – Part 6: Technical delivery conditions for flat products of high yield strength structural steels in the quenched and tempered condition*. NEN. Delft, The Netherlands, 2022.

[43] *NEN-EN 10028-2:2017 – Flat products made of steels for pressure purposes – Part 2: Non-alloy and alloy steels with specified elevated temperature properties*. NEN. Delft, The Netherlands, 2017.

[44] *NEN-EN 10028-3:2017 – Flat products made of steels for pressure purposes – Part 3: Weldable fine grain steels, normalized*. NEN. Delft, The Netherlands, 2017.

[45] *NEN-EN 10028-4:2017 – Flat products made of steels for pressure purposes – Part 4: Nickel alloy steels with specified low temperature properties*. NEN. Delft, The Netherlands, 2017.

[46] *NEN-EN 10028-5:2017 – Flat products made of steels for pressure purposes – Part 5: Weldable fine grain steels, thermomechanically rolled*. NEN. Delft, The Netherlands, 2017.

[47] *NEN-EN 10028-6:2017 – Flat products made of steels for pressure purposes – Part 6: Weldable fine grain steels, quenched and tempered*. NEN. Delft, The Netherlands, 2017.

[48] *NEN-EN 10028-7:2016 – Flat products made of steels for pressure purposes – Part 7: Stainless steels*. NEN. Delft, The Netherlands, 2016.

[49] *NEN-EN 10088-1:2024 – Stainless steels – Part 1: List of stainless steels*. NEN. Delft, The Netherlands, 2024.

[50] *NEN-EN 10088-2:2024 – Stainless steels – Part 2: Technical delivery conditions for sheet/plate and strip of corrosion resistant steels for general purposes*. NEN. Delft, The Netherlands, 2024.

[51] *NEN-EN 10088-3:2024 – Stainless steels – Part 3: Technical delivery conditions for semi-finished products, bars, rods, wire, sections and bright products of corrosion resistant steels for general purposes*. NEN. Delft, The Netherlands, 2024.

[52] *NEN-EN 10088-4:2009 – Stainless steels – Part 4: Technical delivery conditions for sheet/plate and strip of corrosion resisting steels for construction purposes*. NEN. Delft, The Netherlands, 2009.

[53] *NEN-EN 10088-5:2009 – Stainless steels – Part 5: Technical delivery conditions for bars, rods, wire, sections and bright products of corrosion resisting steels for construction purposes*. NEN. Delft, The Netherlands, 2009.

[54] *NEN-EN-ISO 4957:2018 – Tool steels*. NEN. Delft, The Netherlands, 2018.

[55] World Economic Forum. *What is steel scrap and how can it help us reach net zero?* https://www.weforum.org/stories/2023/01/davos23-steel-scrap-decarbonization/. Accessed: 2026-01-05. 2023.

A

Research Paper | Appendix

Starts on the next page

TU Delft report number:
2026.MME.9185

Paper Summary

TU Delft | Mechanical Engineering
Multi-Machine Engineering

Development of a Recipe-based Strategy for Scrap Sorting

Author: E. Le Blansch,[1] supervised by: Prof. dr. ir. D.L. Schott,[1] Dr. ir. Y. Pang[1]
[1]Delft University of Technology, Faculty ME, Mekelweg 2, 2628 CD, Delft, the Netherlands

ARTICLE INFO

Submission date: 22 June 2026

Keywords:
Scrap sorting
Sensor-based sorting
Steel recycling
Recipe-based allocation
Optimisation

ABSTRACT

This appendix presents a paper-style summary of the proposed sensor-based scrap sorting system. The system uses piece-by-piece compositional information from scanned scrap items to construct heaps that satisfy the chemical constraints of high-value steel recipes. The approach combines recipe selection, mass-capacity constraints, and element-wise compositional bounds into an allocation framework. Candidate algorithms are evaluated using indicators such as mass conversion, heap utilization, generated value, and computational performance. The results provide insight into how recipe-based allocation can improve the utilization and value recovery of mixed scrap streams.

I INTRODUCTION

Steel is essential for construction, infrastructure, transport, and manufacturing, but its production is resource- and energy-intensive. In 2024, global crude steel production reached 1 885 million tonnes, more than twice the level reported in 2000 [1]. Steel production is also responsible for approximately 8% of global CO_2 emissions [2]. Increasing the use of recycled scrap is therefore a key route toward lower-emission and more circular steel production. Steel can be recycled repeatedly when impurities are controlled [1], and electric arc furnace (EAF) steelmaking avoids part of the carbon-intensive iron ore reduction step [3].

The main challenge is that end-of-life scrap originates from many products and alloy systems, leading to large variation in the chemical composition of individual scrap pieces [4]. Modern steel grades are defined by strict compositional limits, while the final melt composition depends on the mass-weighted average of all inputs. Residual and tramp elements such as copper and tin, or unsuitable alloying elements, can exceed allowable limits and force dilution with primary material or down-cycling into lower-value products [3, 5].

Sensor-based scrap analysis creates opportunities to use scrap more strategically. XRF, LIBS, and XRT can increasingly support identification, classification, and chemical characterisation of individual scrap pieces [6, 7]. LIBS has also shown potential for high-speed alloy-level classification of post-consumer metal scrap [8]. However, conventional sorting systems still often classify scrap into broad material categories rather than using detailed piece-level composition data for recipe allocation [4].

This paper proposes a recipe-based scrap sorting decision layer. Instead of sorting scrap only by material class, the system evaluates whether combinations of individual scrap items can satisfy the chemical limits of specific steel recipes. A heap is feasible when the mass-weighted average composition of the selected items lies within the recipe bounds while respect-

ing a maximum heap capacity. The contribution of this paper is a simulation framework that combines piece-level scrap data, European Steel Standard recipe constraints, and heuristic allocation algorithms to evaluate recipe-based scrap sorting under varying buffer sizes, heap capacities, and heap counts.

II LITERATURE REVIEW

Scrap steel originates from end-of-life vehicles, demolished buildings, industrial off-cuts, and consumer products [9]. It is an important secondary raw material, especially in regions with high scrap availability [10]. Home and prompt scrap usually have more predictable compositions, while obsolete scrap is more heterogeneous [11]. This heterogeneity limits high-grade recycling because residual elements such as copper and tin are difficult to remove during melting and can reduce ductility, formability, or surface quality [3, 12]. Well-sorted scrap therefore has higher value and broader usability [5]. Traditional magnetic and eddy-current sorting are robust and fast, but they do not provide the chemical detail required for recipe-based production [13, 14].

Magnetic, eddy-current, and induction-based sorting are suitable for bulk separation, but provide limited alloy-level information [13, 14, 15]. XRT distinguishes materials mainly through density-related X-ray attenuation, while XRF provides element-specific information but involves a trade-off between accuracy, cost, and throughput [16, 7]. LIBS is especially relevant for alloy-level sorting because it can provide detailed elemental information and detect lighter elements, although it remains sensitive to surface condition, calibration, and preprocessing [8]. Optical and NIR systems can classify shape, coatings, and surface characteristics, but cannot directly determine elemental composition [17, 18].

Commercial systems from TOMRA [19], SGM Magnetics [20, 21], STEINERT [22], and Ocean Optics/Austin AI [23] demonstrate that sensor-based alloy sorting can reach purities above 95% and throughputs of several tonnes per hour, depending on system configuration and input material. However, most applications classify items into predefined material categories, especially for aluminium and non-ferrous scrap. They do not decide how individual ferrous scrap items should be combined to satisfy steel recipe constraints.

Previous work has addressed scrap sorting through optimisation, sensor classification, process simulation, and vision-based inspection. Bernatzki et al. [24] formulated scrap selection as a mixed-integer programming problem using predefined scrap categories. Compañero et al. [4] showed that better compositional information can reduce production costs, but did not develop a piece-level heap allocation method. Wu et al. [25] modelled automated aluminium sorting using DEM, while Williams et al. [26], Díaz-Romero et al. [27], and Xu et al. [18] focused on sensor- or vision-based classification.

These studies show that sensing, classification, and optimisation can improve scrap sorting. However, they generally classify items, optimise physical flow, or use aggregated scrap classes. The remaining gap is a decision layer that uses piece-level mass and composition data to combine scrap items into heaps whose weighted-average composition satisfies specific steel recipes.

III METHODOLOGY

A Sorting Decision Layer

The proposed system is a decision layer on top of a sensor-based scrap sorting process. Scrap items are scanned for mass and chemical composition, stored in a buffer, and assigned to recipe-compliant heaps. Each heap corresponds to a target recipe and is accepted only if its total mass and mass-weighted elemental composition satisfy the recipe constraints.

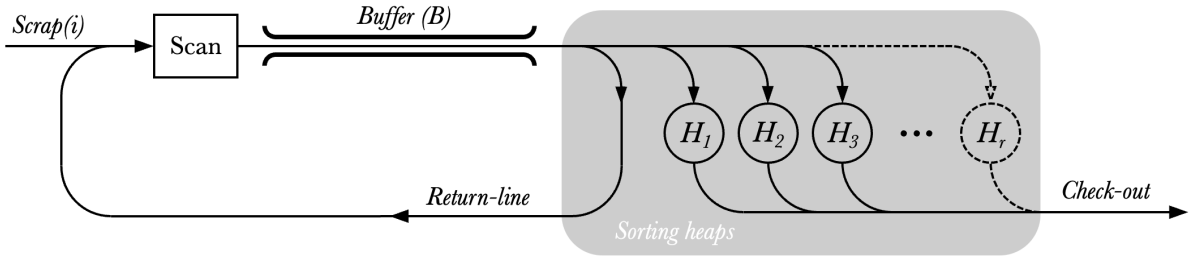


Figure A.1: Material flow of the proposed recipe-based scrap sorting system.

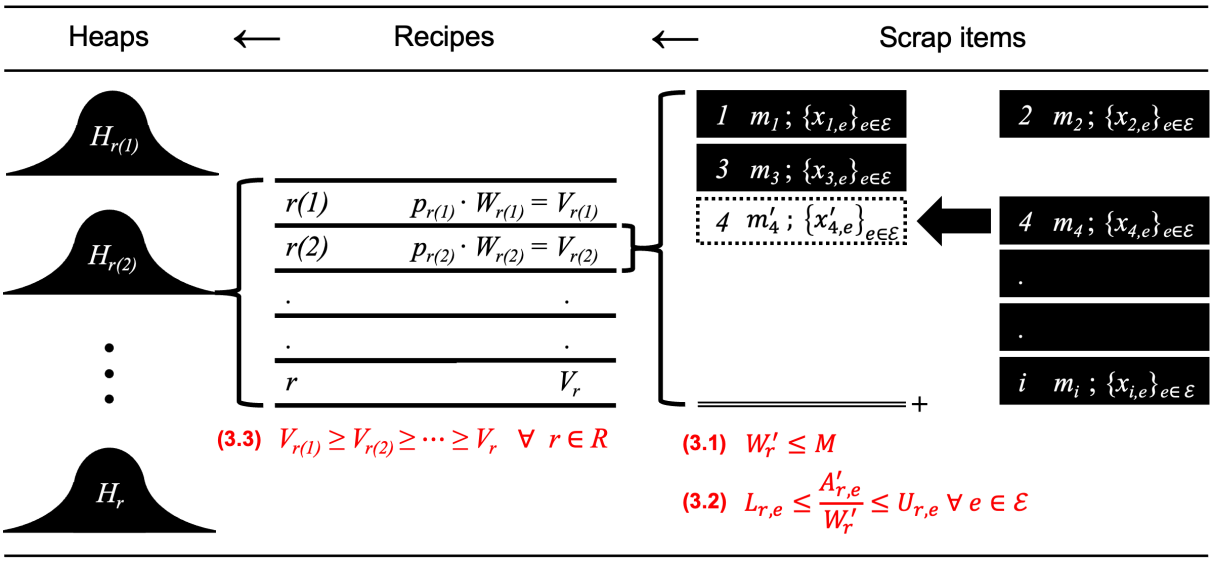


Figure A.2: Conceptual representation of the recipe-based heap allocation model.

In each allocation round, candidate heaps are constructed for all available recipes. The highest-value feasible heap is finalised, its scrap items are removed from the buffer, and the next round starts. This continues until the maximum number of heaps is reached or no feasible heap can be formed.

Let $i \in \mathcal{S}$ denote scrap items, $r \in \mathcal{R}$ recipes, and $e \in \mathcal{E}$ elements. Each item has mass m_i and composition $x_{i,e}$. Recipe bounds are denoted by $L_{r,e}$ and $U_{r,e}$, and heap capacity by M. A heap H_r is feasible if:

$$\sum_{i\in H_r} m_i \leq M. \tag{A.1}$$

$$L_{r,e} \leq \frac{\sum_{i\in H_r} m_i x_{i,e}}{\sum_{i\in H_r} m_i} \leq U_{r,e}, \quad \forall e \in \mathcal{E}. \tag{A.2}$$

The value of a candidate heap is:

$$V_r = p_r \sum_{i\in H_r} m_i, \tag{A.3}$$

where p_r is the recipe price per kilogram. During heap construction, an item is rejected if it is already assigned, exceeds capacity, or violates any updated elemental bound.

B Candidate Algorithms

For one heap, the search space scales as $R \times 2^{\mathcal{S}}$, making exhaustive enumeration infeasible for 315 recipes and large buffers. Exact methods such as ILP, MILP, branch-and-bound, and branch-and-cut can provide optimality guarantees for small instances, but scale poorly [28, 29, 30]. Therefore, greedy heuristics were evaluated.

Five variants were tested: single-run greedy, multi-pass greedy, multi-pass greedy with local search, multi-start greedy, and multi-start greedy with local search. Single-run greedy attempts one mass-sorted pass. Multi-pass greedy repeatedly scans the remaining items until no further feasible addition is possible. Multi-start greedy repeats greedy construction using randomised orders. Local-search variants attempt item swaps followed by greedy refilling. For each round, the best candidate heap over all recipes is selected using Equation (A.3).

C Simulation Data and Experiments

Because large-scale public datasets with piece-level scrap compositions are unavailable, synthetic scrap data were generated from European steel recipe data. The recipe database was compiled from European standards for structural steels, pressure-vessel steels, stainless steels, and tool steels [31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47]. It contains 315 recipes with elemental minimum and maximum limits. Where multiple product configurations were available, restrictive variants were selected, while derived constraints such as carbon equivalent values were excluded.

Synthetic scrap items were generated by sampling recipe rows with replacement and drawing each tracked element uniformly between its lower and upper bounds. Each item was assigned a random mass and treated as chemically uniform and indivisible. Sensor uncertainty, contamination, coatings, and oxidation were not modelled, because the aim was to evaluate the allocation strategy rather than sensor error.

Algorithm	Item conversion	Mass conversion	Value	Compute time
Single-run Greedy	33%	24%	€84 168,21	23 359 ms
Multi-pass Greedy	50%	49%	€175 218,18	65 223 ms
Multi-pass Greedy + local search	50%	49%	€175 218,18	114 316 ms
Multi-start Greedy	61%	51%	€183 142,54	636 020 ms
Multi-start Greedy + local search	61%	51%	€181 314,47	846 035 ms

Table A.1: Performance comparison of candidate algorithms.

The simulation was implemented in VBA in Microsoft Excel to allow transparent inspection of intermediate results. Performance was evaluated using item conversion, mass conversion, heap utilisation, economic value, and throughput. The experiments compared candidate algorithms, assessed buffer size, and then varied heap capacity up to 20 000 kg and heap count up to 12. Finally, economic value was multiplied by heap utilisation to identify balanced configurations.

IV RESULTS

A Algorithm Performance

Table A.1 shows that the single-run greedy algorithm gives the lowest conversion and value. Multi-pass greedy improves mass conversion from 24% to 49% and value from €84 168,21 to €175 218.18. Adding local search does not improve the solution but increases runtime. Multi-start greedy gives the highest value (€183 142.54), but only 4.5% above multi-pass greedy while requiring almost ten times more computation. Therefore, multi-pass greedy was selected as the baseline algorithm.

B Buffer Size Assessment

Figure A.3 shows that performance improves rapidly at small buffer sizes because more scrap items increase compositional diversity. From 50 to 300 items, item conversion rises from 18% to 48%, and mass conversion from 16% to 38%. Beyond approximately 700-1 000 items, improvements become smaller and fluctuate around a plateau. A buffer size of 1000 items was therefore selected as a stable baseline.

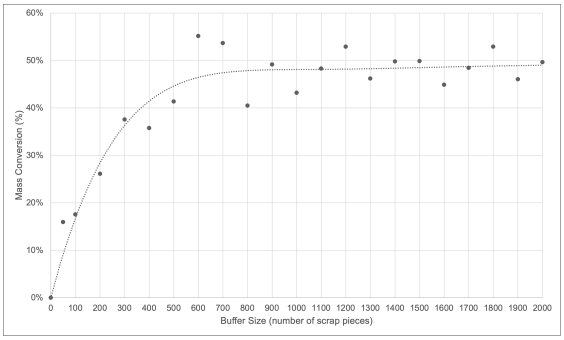


Figure A.3: Cumulative mass conversion as a function of buffer size.

C Mass Conversion

The selected multi-pass greedy algorithm was tested under varying heap capacities and heap counts. Figure A.4 shows that mass conversion increases with both parameters. At 1 000 kg, conversion ranges from 1% with one heap to 12% with twelve heaps. Increasing capacity from 2000 kg to 6000 kg raises conversion from 12% to 37% for six heaps and from 25% to 68% for twelve heaps. Beyond approximately 8 000-10 000 kg, improvements diminish, indicating saturation of feasible combinations.

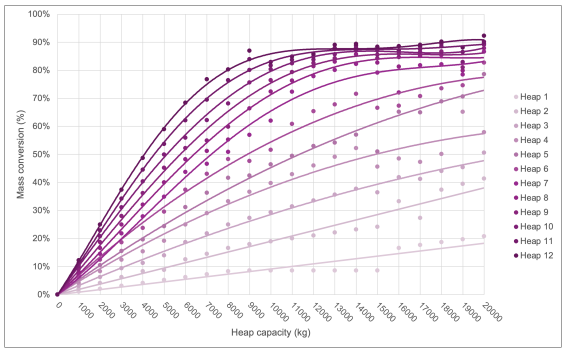


Figure A.4: Cumulative mass conversion for different heap capacities and numbers of heaps.

D Heap Utilisation

Heap utilisation shows the opposite trend. As shown in Figure A.5, small heaps are filled almost completely, but utilisation decreases at larger capacities. With twelve heaps, utilisation falls from 98% at 1 000 kg to 66% at 10 000 kg and 37% at 20 000 kg. Larger heaps can convert more mass, but are harder to fill completely while satisfying the chemical constraints.

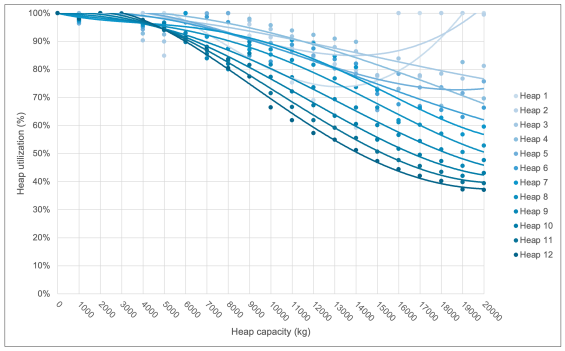


Figure A.5: Cumulative heap utilisation for different heap capacities and numbers of heaps.

E Economic Value

Economic value follows the mass-conversion trend. In Figure A.6, value increases strongly at lower capacities because more material can be assigned to recipe-compliant heaps. For twelve heaps, value increases from €60 178 at 1000 kg to €211 388 at 6 000 kg. However, larger capacities provide diminishing gains; from 8 000 kg to 20 000 kg the increase is only €8 840 for twelve heaps. This shows that the most profitable feasible combinations are mostly captured at intermediate capacities.

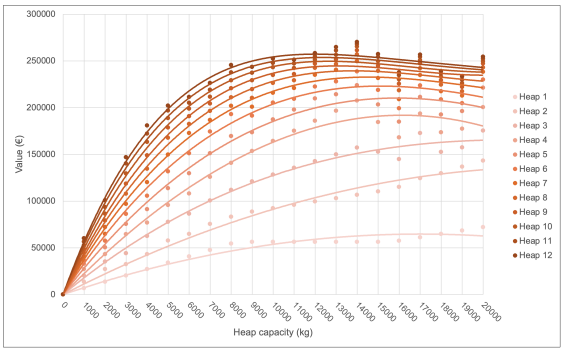


Figure A.6: Cumulative economic value for different heap capacities and numbers of heaps.

F Value and Heap Utilisation Trade-off

A utilisation-weighted value score was calculated by multiplying economic value by heap utilisation. Figure A.7 shows that performance increases from small to intermediate heap capacities, then declines when utilisation losses outweigh value gains. The best configuration is seven heaps with a heap capacity of 12 000 kg, reaching a maximum score of 209 653. This configuration balances sufficient combinatorial flexibility with efficient heap filling.

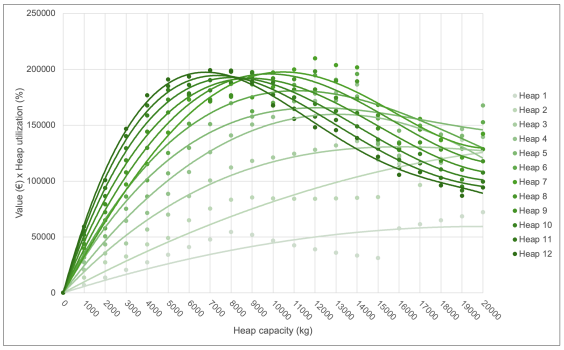


Figure A.7: Utilisation-weighted value score for different heap capacities and numbers of heaps.

For this selected configuration, Figure A.8 shows that scrap is distributed across several recipes rather than concentrated in one output stream. This indicates that recipe diversity improves conversion and value.

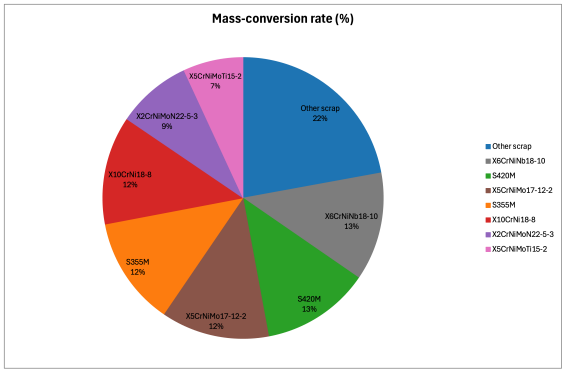


Figure A.8: Mass-conversion distribution for the selected configuration with seven heaps and a heap capacity of 12 000 kg per heap.

Figure A.9 illustrates the chemical feasibility of one selected heap. Individual items may fall outside recipe limits for some elements, but the mass-weighted average remains within the allowable range. This demonstrates the central principle of the method: complementary scrap pieces can be combined into feasible, higher-value recipes.

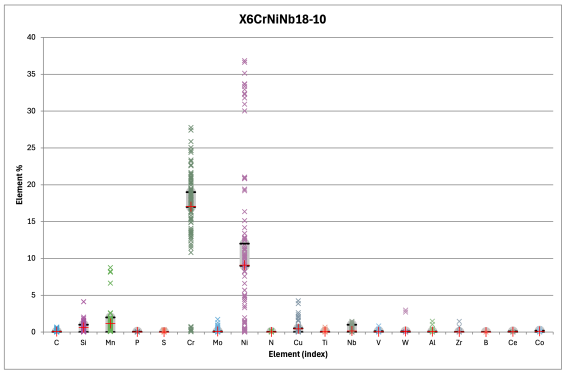


Figure A.9: Elemental composition of a selected heap assigned to recipe X6CrNiNb18-10. Points represent individual scrap items, black bars indicate recipe limits, and crosses indicate the mass-weighted average composition.

V DISCUSSION

The results show that a recipe-based decision layer can translate piece-level scrap composition data into value-oriented steel recipe allocations. The multi-pass greedy algorithm provides a practical compromise between performance and runtime: it achieves near-best value while avoiding the high computational cost of multi-start variants. This makes it suitable for repeated simulations and potential decision-support use.

The experiments also show that system tuning is as important as algorithm selection. Buffer size improves performance until sufficient compositional diversity is available, after which additional items mainly increase computational effort. Similarly, increasing heap capacity and heap count improves mass conversion and value, but reduces heap utilisation at larger capacities.

The best-balanced setting, seven heaps of 12000 kg, achieves approximately 78% mass conversion and 89% heap utilisation.

Industrial implementation is most realistic as a batch-based decision-support layer connected to existing sensor-based sorting systems. Sensors provide item-level mass and composition data, while the allocation model periodically assigns buffered items to recipe-compliant heaps. The greedy algorithm can also be simplified by limiting passes or preselecting candidate recipes, which supports scalability under changing scrap streams, prices, and production priorities.

Several limitations remain. The simulation assumes accurate item mass and composition, while real sensors are affected by noise, surface contamination, coatings, oxidation, and incomplete elemental coverage. The scrap data are synthetic and may not capture correlations, mass distributions, contamination, or temporal patterns in real industrial streams. The recipe model also omits industrial constraints such as carbon equivalent limits, alloy correction, furnace scheduling, order timing, and logistics. Finally, the greedy algorithm does not guarantee global optimality and may miss better combinations in constrained scrap pools.

Compared with conventional sorting, the proposed method changes the sorting objective. Instead of separating material into broad categories, it allocates scrap to steel recipes based on weighted-average chemical feasibility. This allows items with unfavourable individual compositions to be used when balanced by complementary items. The method therefore shifts scrap sorting from material classification toward value-oriented recipe allocation.

VI CONCLUSION

This paper developed and evaluated a recipe-based scrap sorting decision layer for high-value steel recycling. The system combines piece-level scrap information, recipe constraints from European Steel Standards, and heuristic allocation algorithms to form recipe-compliant heaps based on mass-weighted chemical composition.

The results show that sensor-based composition data can be used not only for classification, but also for strategic allocation. Among the tested algorithms, multi-pass greedy provides the best balance between solution quality and computation time. Buffer-size experiments show diminishing returns beyond approximately 700-1000 items, and system-level experiments reveal a trade-off between mass conversion, heap utilisation, and economic value. The best-balanced configuration was seven heaps with a heap capacity of 12000 kg, achieving approximately 78% mass conversion and 89% heap utilisation.

The proposed approach demonstrates that complementary scrap pieces can be combined into chemically feasible and economically valuable steel recipes, even when individual pieces fall outside some recipe limits. This supports a shift from separation-oriented scrap sorting toward value-oriented recipe allocation.

The findings should be interpreted as a proof of concept rather than a direct industrial prediction. Future work should validate the framework with real piece-level scrap data, include sensor uncertainty, refine recipe and production constraints, and add logistical limitations such as conveyor capacity, storage layout, pile accessibility, and furnace scheduling. Despite these limitations, the framework provides a practical basis for integrating sensor-based scrap characterisation with recipe-level steel production decisions.

References

[1] World Steel Association. World Steel in Figures 2025. Statistical Report. Accessed: 26 January 2026. World Steel Association, 2025. URL: https://worldsteel.org/wp-content/uploads/World-Steel-in-Figures-2025.pdf.

[2] Dierk Raabe et al. “Circular steel for fast decarbonization: thermodynamics, kinetics, and microstructure behind upcycling scrap into high-performance sheet steel”. In: Annual review of materials research 54.1 (2024), pp. 247–297.

[3] Ishwar Kapoor, Claire Davis, and Zushu Li. “Effects of residual elements during the casting process of steel production: a critical review”. In: Ironmaking & steelmaking 48.6 (2021), pp. 712–727.

[4] Reinol Josef Compañero et al. “Appraising the value of compositional information and its implications to scrap-based production of steel”. In: Mineral Economics 36.3 (2023), pp. 463–480.

[5] Beatriz Pérez Horno, Andreas Feldmann, and Peter Samuelsson. “Mapping the dynamics shaping the future of steel recycling”. In: Discover Sustainability 6.1 (2025), p. 808.

[6] Leslie Brooks and Gabrielle Gaustad. “The potential for XRF & LIBS handheld analyzers to perform material characterization in scrap yards”. In: Journal of Sustainable Metallurgy 7.2 (2021), pp. 732–754.

[7] Max Kölking et al. “More resource efficient recycling of copper and copper alloys by using X-ray fluorescence sorting systems: An investigation on the metallic fraction of mixed foundry residues”. In: Waste Management & Research 42.9 (2024), pp. 814–822.

[8] Dillam Jossue Díaz-Romero et al. “Real-time classification of aluminum metal scrap with laser-induced breakdown spectroscopy using deep and other machine learning approaches”. In: Spectrochimica Acta Part B: Atomic Spectroscopy 196 (2022), p. 106519.

[9] Waseem S Khan et al. “Recycling and reusing of engineering materials: Recycling for sustainable developments”. In: (2022).

[10] Takuma Watari et al. “Global stagnation and regional variations in steel recycling”. In: Resources, Conservation and Recycling 220 (2025), p. 108363.

[11] Bo Björkman and Caisa Samuelsson. “Recycling of steel”. In: Handbook of recycling. Elsevier, 2014, pp. 65–83.

[12] Ishwar Kapoor, Claire Davis, and Zushu Li. “Effect of Residual Elements during the Hot-Working Process of Steel Production: A Critical Review”. In: steel research international 95.9 (2024), p. 2400116.

[13] J Svoboda and T Fujita. “Recent developments in magnetic methods of material separation”. In: Minerals Engineering 16.9 (2003), pp. 785–792.

[14] York R Smith, James R Nagel, and Raj K Rajamani. “Eddy current separation for recovery of non-ferrous metallic particles: A comprehensive review”. In: Minerals Engineering 133 (2019), pp. 149–159.

[15] Kane C Williams, Michael D O’Toole, and Anthony J Peyton. “Scrap metal classification using magnetic induction spectroscopy and machine vision”. In: IEEE Transactions on Instrumentation and Measurement 72 (2023), pp. 1–11.

[16] Christopher Robben et al. “X-ray-transmission based ore sorting at the San Rafael tin mine”. In: Minerals Engineering 145 (2020), p. 105870.

[17] Mary Cesetti and Piergiorgio Nicolosi. “Waste processing: new near infrared technologies for material identification and selection”. In: Journal of Instrumentation 11.09 (2016), p. C09002.

[18] Wenguang Xu et al. “Classification and rating of steel scrap using deep learning”. In: Engineering applications of artificial intelligence 123 (2023), p. 106241.

[19] TOMRA Recycling. AUTOSORT™ PULSE: Dynamic LIBS Sorting System. Accessed: 2025-12-16. URL: https://www.tomra.com/waste-metal-recycling/products/machines/autosort-pulse.

[20] SGM Magnetics. X-ray Separators: XRT and XRF Sorting Systems. Accessed: 2025-12-16. URL: https://www.sgmmagnetics.com/en/products/x-ray-separators-xrt-xrf-t/.

[21] SGM Magnetics. LIBS Separator Cleansort R. https://www.sgmmagnetics.com/en/products/libs-separator-cleansort-r/. Accessed: 2025-12-16. 2025.

[22] STEINERT GmbH. STEINERT PLASMAX: LIBS Sorting System for Metal Recycling. Accessed: 2025-12-16. URL: https://steinertglobal.com/us/sorting-systems/sensor-sorting/libs-sorting-systems/steinert-plasmax-libs/.

[23] Ocean Optics. High-Throughput Sensing System for Precision Scrap Metal Sorting. Accessed: 2025-12-16. URL: https://www.oceanoptics.com/metal-recycling/high-throughput-sensing-system-provides-precision-sorting-of-scrap-aluminum/.

[24] Klaus-P Bernatzki et al. “Optimal scrap combination for steel production”. In: Operations-Research-Spektrum 20.4 (1998), pp. 251–258.

[25] Yongli Wu, Tijmen Oudshoorn, and Peter Rem. “Modelling and optimization of an innovative facility for automated sorting of aluminium scraps”. In: Waste Management 189 (2024), pp. 103–113.

[26] Kane C Williams et al. “Classification of shredded aluminium scrap metal using magnetic induction spectroscopy”. In: Sensors 23.18 (2023), p. 7837.

[27] Dillam Diaz-Romero et al. “Classification of aluminum scrap by laser induced breakdown spectroscopy (LIBS) and RGB+ D image fusion using deep learning approaches”. In: Resources, Conservation and Recycling 190 (2023), p. 106865.

[28] George L Nemhauser and Laurence A Wolsey. Integer and combinatorial optimization. Vol. 18. Wiley New York, 1988.

[29] David R Morrison et al. “Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning”. In: Discrete optimization 19 (2016), pp. 79–102.

[30] John E Mitchell. “Branch-and-cut algorithms for combinatorial optimization problems”. In: Handbook of applied optimization 1.1 (2002), pp. 65–77.

[31] NEN-EN 10025-2:2019 – Hot rolled products of structural steels – Part 2: Technical delivery conditions for non-alloy structural steels. NEN. Delft, The Netherlands, 2019.

[32] NEN-EN 10025-3:2019 – Hot rolled products of structural steels – Part 3: Technical delivery conditions for normalized/normalized rolled weldable fine grain structural steels. NEN. Delft, The Netherlands, 2019.

[33] NEN-EN 10025-4:2019+A1:2022 – Hot rolled products of structural steels – Part 4: Technical delivery conditions for thermomechanical rolled weldable fine grain structural steels. NEN. Delft, The Netherlands, 2022.

[34] NEN-EN 10025-5:2019 – Hot rolled products of structural steels – Part 5: Technical delivery conditions for structural steels with improved atmospheric corrosion resistance. NEN. Delft, The Netherlands, 2019.

[35] NEN-EN 10025-6:2019+A1:2022 – Hot rolled products of structural steels – Part 6: Technical delivery conditions for flat products of high yield strength structural steels in the quenched and tempered condition. NEN. Delft, The Netherlands, 2022.

[36] NEN-EN 10028-2:2017 – Flat products made of steels for pressure purposes – Part 2: Non-alloy and alloy steels with specified elevated temperature properties. NEN. Delft, The Netherlands, 2017.

[37] NEN-EN 10028-3:2017 – Flat products made of steels for pressure purposes – Part 3: Weldable fine grain steels, normalized. NEN. Delft, The Netherlands, 2017.

[38] NEN-EN 10028-4:2017 – Flat products made of steels for pressure purposes – Part 4: Nickel alloy steels with specified low temperature properties. NEN. Delft, The Netherlands, 2017.

[39] NEN-EN 10028-5:2017 – Flat products made of steels for pressure purposes – Part 5: Weldable fine grain steels, thermomechanically rolled. NEN. Delft, The Netherlands, 2017.

[40] NEN-EN 10028-6:2017 – Flat products made of steels for pressure purposes – Part 6: Weldable fine grain steels, quenched and tempered. NEN. Delft, The Netherlands, 2017.

[41] NEN-EN 10028-7:2016 – Flat products made of steels for pressure purposes – Part 7: Stainless steels. NEN. Delft, The Netherlands, 2016.

[42] NEN-EN 10088-1:2024 – Stainless steels – Part 1: List of stainless steels. NEN. Delft, The Netherlands, 2024.

[43] NEN-EN 10088-2:2024 – Stainless steels – Part 2: Technical delivery conditions for sheet/plate and strip of corrosion resistant steels for general purposes. NEN. Delft, The Netherlands, 2024.

[44] NEN-EN 10088-3:2024 – Stainless steels – Part 3: Technical delivery conditions for semi-finished products, bars, rods, wire, sections and bright products of corrosion resistant steels for general purposes. NEN. Delft, The Netherlands, 2024.

[45] NEN-EN 10088-4:2009 – Stainless steels – Part 4: Technical delivery conditions for sheet/plate and strip of corrosion resisting steels for construction purposes. NEN. Delft, The Netherlands, 2009.

[46] NEN-EN 10088-5:2009 – Stainless steels – Part 5: Technical delivery conditions for bars, rods, wire, sections and bright products of corrosion resisting steels for construction purposes. NEN. Delft, The Netherlands, 2009.

[47] NEN-EN-ISO 4957:2018 – Tool steels. NEN. Delft, The Netherlands, 2018.

B

Excel Tables | Appendix

This appendix provides a visual overview of the Microsoft Excel implementation used for the data-generation and allocation procedure. The screenshots are included to clarify how the main intermediate and output tables were structured in the spreadsheet environment. Figure B.1 shows the input recipe table containing the steel grades, source standards, assigned prices, and elemental composition limits. Figure B.2 shows the candidate scrap table generated by sampling recipe rows from the input recipe table, while Figure B.3 shows the resulting generated scrap-item dataset with elemental compositions and assigned masses. The allocation output is illustrated in Figure B.4, which shows the candidate-heap result tables created for successive allocation rounds. Finally, Figure B.5 shows the final allocation and annotation table, in which generated scrap items are linked to their assigned recipes and corresponding values. The screenshots are intended to illustrate the spreadsheet structure rather than to present the complete datasets, as several tables extend beyond the visible image frame.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Mat. no.	Symbol	Norm	Price (€/kg)	C % min.	C % max.	Si % min.	Si % max.	Mn % min.	Mn % max.	P % min.	P % max.	S % min.	S % max.	Cr % min.	Cr % max.	Mo % min.	Mo % max.	Ni % min.	Ni % max.
2	10035	S185	EN 10025	0,67	0	0,17	0	0	0	1,4	0	0,045	0	0,045	0	0	0	0	0	0
3	10038	S235JR	EN 10025	0,67	0	0,17	0	0,5	0	1,4	0	0,035	0	0,035	0	0,29	0	0,11	0	0,42
4	10044	S275JR	EN 10025	0,67	0	0,21	0	0,5	0	1,5	0	0,035	0	0,035	0	0,29	0	0,11	0	0,42
5	10045	S355JR	EN 10025	0,67	0	0,24	0	0,55	0	1,6	0	0,035	0	0,035	0	0,29	0	0,11	0	0,42
6	10050	E295	EN 10025	0,67	0	0,1	0	0,5	0	1,5	0	0,045	0	0,045	0	0,5	0	0,1	0	0,5
7	10060	E335	EN 10025	0,67	0	0,1	0	0,5	0	1,5	0	0,045	0	0,045	0	0,5	0	0,1	0	0,5
8	10070	E360	EN 10025	0,67	0	0,1	0	0,5	0	1,5	0	0,045	0	0,045	0	0,5	0	0,1	0	0,5
9	10114	S235J0	EN 10025	0,67	0	0,17	0	0,5	0	1,4	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
10	10117	S235J2	EN 10025	0,67	0	0,17	0	0,5	0	1,4	0	0,025	0	0,025	0	0,29	0	0,11	0	0,42
11	10143	S275J0	EN 10025	0,67	0	0,18	0	0,5	0	1,5	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
12	10145	S275J2	EN 10025	0,67	0	0,18	0	0,5	0	1,5	0	0,025	0	0,025	0	0,29	0	0,11	0	0,42
13	10345	P235GH	EN 10028	0,67	0	0,16	0	0,35	0,6	1,2	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
14	10425	P265GH	EN 10028	0,67	0	0,2	0	0,4	0,8	1,4	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
15	10473	P355GH	EN 10028	0,67	0,1	0,22	0	0,6	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
16	10481	P295GH	EN 10028	0,67	0,08	0,2	0	0,4	0,9	1,5	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
17	10487	P275NH	EN 10028	0,67	0	0,16	0	0,4	0,8	1,5	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
18	10488	P275NL1	EN 10028	0,67	0	0,16	0	0,4	0,8	1,5	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
19	10490	S275N	EN 10025	0,67	0	0,18	0	0,4	0,5	1,5	0	0,03	0	0,025	0	0,3	0	0,1	0	0,3
20	10491	S275NL	EN 10025	0,67	0	0,16	0	0,4	0,5	1,5	0	0,025	0	0,02	0	0,3	0	0,1	0	0,3
21	10502	S500J0	EN 10025	1,05	0	0,2	0	0,55	0	1,7	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
22	10507	S460JR	EN 10025	1,05	0	0,2	0	0,55	0	1,7	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
23	10538	S460J0	EN 10025	1,05	0	0,2	0	0,55	0	1,7	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
24	10545	S355N	EN 10025	0,67	0	0,2	0	0,5	0,9	1,65	0	0,03	0	0,025	0	0,3	0	0,1	0	0,5
25	10546	S355NL	EN 10025	0,67	0	0,18	0	0,5	0,9	1,65	0	0,025	0	0,02	0	0,3	0	0,1	0	0,5
26	10552	S460J2	EN 10025	1,05	0	0,2	0	0,55	0	1,7	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
27	10553	S355J0	EN 10025	0,67	0	0,2	0	0,55	0	1,6	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
28	10562	P355N	EN 10028	0,67	0	0,18	0	0,5	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
29	10565	P355NH	EN 10028	0,67	0	0,18	0	0,5	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
30	10566	P355NL1	EN 10028	0,67	0	0,18	0	0,5	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
31	10577	S355J2	EN 10025	0,67	0	0,2	0	0,55	0	1,6	0	0,025	0	0,025	0	0,29	0	0,11	0	0,42
32	10581	S460K2	EN 10025	1,05	0	0,2	0	0,55	0	1,7	0	0,03	0	0,03	0	0,29	0	0,11	0	0,42
33	10596	S355K2	EN 10025	0,67	0	0,2	0	0,55	0	1,6	0	0,025	0	0,025	0	0,29	0	0,11	0	0,42
34	11104	P275NL2	EN 10028	0,67	0	0,16	0	0,4	0,8	1,5	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
35	11106	P355NL2	EN 10028	0,67	0	0,18	0	0,5	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
36	11520	C70U	EN ISO 4957	2,8	0,65	0,75	0	0,5	0,5	0,8	0	0,03	0	0,03	0	0,5	0	0,1	0	0,5
37	11525	C80U	EN ISO 4957	2,8	0,75	0,85	0	0,5	0,5	0,8	0	0,03	0	0,03	0	0,5	0	0,1	0	0,5
38	11535	C90U	EN ISO 4957	2,8	0,85	0,95	0	0,5	0,5	0,8	0	0,03	0	0,03	0	0,5	0	0,1	0	0,5
39	11545	C105U	EN ISO 4957	2,8	0,95	1,1	0	0,5	0,3	0,6	0	0,03	0	0,03	0	0,5	0	0,1	0	0,5
40	11555	C120U	EN ISO 4957	2,8	1,1	1,25	0	0,5	0,3	0,6	0	0,03	0	0,03	0	0,5	0	0,1	0	0,5
41	11730	C45U	EN ISO 4957	2,8	0,42	0,5	0	0,5	0,5	0,8	0	0,03	0	0,03	0	0,5	0	0,1	0	0,5
42	14000	X6Cr13	EN 10088	2,2	0	0,08	0	1	0	1	0	0,04	0	0,015	12	14	0	0,1	0	0,5

Table1 Table2 Table3 Table4 Table5 +

Figure B.1: Screenshot of the input recipe table $\mathcal{T}_1$ in Microsoft Excel. The table contains the steel grade, source standard, assigned price, and minimum and maximum elemental composition limits used for scrap data generation. The recipes are based on the European steel standards listed in Table 5.1. The screenshot is included to illustrate the spreadsheet structure, only part of the full table is visible, as the table extends beyond the image frame.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Mat. no.	Symbol	Norm	Price (€/kg)	C % min.	C % max.	Si % min.	Si % max.	Mn % min.	Mn % max.	P % min.	P % max.	S % min.	S % max.	Cr % min.	Cr % max.	Mo % min.	Mo % max.	Ni % min.	Ni % max.
2	10481	P295GH	EN 10028	0,67	0,08	0,2	0	0,4	0,9	1,5	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
3	14303	X4CrNi18-12	EN 10088	3,6	0	0,06	0	1	0	2	0	0,045	0	0,015	17	19	0	0,1	11	13
4	14646	X6CrMnNiCuN18-12-4-2	EN 10088	3,6	0,02	0,1	0	1	10,5	12,5	0	0,05	0	0,015	17	19	0	0,5	3,5	4,5
5	14477	X2CrNiMoN29-7-2	EN 10088	6,8	0	0,03	0	0,5	0,8	1,5	0	0,03	0	0,015	28	30	1,5	2,6	5,8	7,5
6	14373	X12CrMnNiN18-9-5	EN 10088	3,6	0	0,15	0	1	7,5	10,5	0	0,045	0	0,015	17	19	0	0,1	4	6
7	14372	X12CrMnNiN17-7-5	EN 10088	3,6	0	0,15	0	1	5,5	7,5	0	0,045	0	0,015	16	18	0	0,1	3,5	5,5
8	14378	X6CrMnNiN18-13-3	EN 10088	2,2	0	0,08	0	1	11,5	14,5	0	0,06	0	0,03	17	19	0	0	2,3	3,7
9	14941	X6CrNiTiB18-10	EN 10088	3,6	0,04	0,08	0	1	0	2	0	0,035	0	0,015	17	19	0	0,1	9	12
10	18943	S420J0W	EN 10025	0,8	0	0,2	0	0,65	0,5	1,35	0	0,035	0	0,035	0,4	0,8	0	0,3	0	0,65
11	18870	P460Q	EN 10028	0,67	0	0,18	0	0,5	0	1,7	0	0,025	0	0,01	0	0,5	0	0,5	0	1
12	14028	X30Cr13	EN 10088	2,2	0,26	0,36	0	1	0	1,5	0	0,04	0	0,015	12	14	0	0,1	0	0,5
13	14560	X3CrNiCu19-9-2	EN 10088	3,6	0	0,035	0	1	1,5	2	0	0,045	0	0,015	18	19	0	0,1	8	9
14	18867	P355QH	EN 10028	0,67	0	0,16	0	0,4	0	1,5	0	0,025	0	0,01	0	0,3	0	0,25	0	0,5
15	14945	X6CrNiWNbN16-16	EN 10088	4	0,04	0,1	0,3	0,6	0	1,5	0	0,035	0	0,015	15,5	17,5	0	0,1	15,5	17,5
16	14028	X30Cr13	EN 10088	2,2	0,26	0,36	0	1	0	1,5	0	0,04	0	0,015	12	14	0	0,1	0	0,5
17	14507	X2CrNiMoCuN25-6-3	EN 10088	6,8	0	0,03	0	0,7	0	2	0	0,035	0	0,015	24	26	3	4	6	8
18	14571	X6CrNiMoTi17-12-2	EN 10088	3,6	0	0,08	0	1	0	2	0	0,045	0	0,015	16,5	18,5	2	2,5	10,5	13,5
19	18869	P355QL2	EN 10028	0,67	0	0,16	0	0,4	0	1,5	0	0,025	0	0,01	0	0,3	0	0,25	0	0,5
20	18901	S460N	EN 10025	1,05	0	0,2	0	0,6	1	1,7	0	0,03	0	0,025	0	0,3	0	0,1	0	0,8
21	14597	X8CrMnCuN17-8-3	EN 10088	3,6	0	0,1	0	2	6,5	9	0	0,04	0	0,03	15	18	0	1	0	3
22	14423	X1CrNiMoCu12-7-3	EN 10088	2,2	0	0,02	0	0,5	0	2	0	0,04	0	0,003	11	13	2,3	2,8	0	0,5
23	18946	S355J2WP	EN 10025	0,8	0	0,12	0	0,75	0	1	0,06	0,15	0	0,035	0,3	1,25	0	0,3	0	0,65
24	18913	P420NL2	EN 10028	0,67	0	0,2	0	0,6	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,1	0	0,8
25	10345	P235GH	EN 10028	0,67	0	0,16	0	0,35	0,6	1,2	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
26	14301	X5CrNi18-10	EN 10088	3,6	0	0,07	0	1	0	2	0	0,045	0	0,015	17,5	19,5	0	0,1	8	10,5
27	18915	P460NL1	EN 10028	0,67	0	0,2	0	0,6	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,1	0	0,8
28	14105	X6CrMoS17	EN 10088	2,2	0	0,08	0	1,5	0	1,5	0	0,04	0,15	0,35	16	18	0,2	0,6	0	0,5
29	18828	P420ML2	EN 10028	0,67	0	0,16	0	0,5	0	1,7	0	0,025	0	0,01	0	0,5	0	0,2	0	0,5
30	14382	X4CrNiCu18-7	EN 10088	3,6	0	0,05	0	1	0	2	0	0,045	0	0,015	17	19	0	0,1	6	8
31	14983	X6CrNiMoTiB17-13	EN 10088	3,6	0,04	0,08	0	0,75	0	2	0	0,035	0	0,015	16	18	2	2,5	12	14
32	10487	P275NH	EN 10028	0,67	0	0,16	0	0,4	0,8	1,5	0	0,025	0	0,01	0	0,3	0	0,08	0	0,5
33	14981	X8CrNiMoNb16-16	EN 10088	4	0,04	0,1	0,3	0,6	0	1,5	0	0,035	0	0,015	15,5	17,5	1,6	2	15,5	17,5
34	14592	X2CrMoTi29-4	EN 10088	2,2	0	0,025	0	1	0	1	0	0,03	0	0,01	28	30	3,5	4,5	0	0,5
35	18823	S355M	EN 10025	1,05	0	0,14	0	0,5	0	1,6	0	0,025	0	0,025	0	0,3	0	0,1	0	0,5
36	14060	X34CrN16	EN 10088	2,2	0,29	0,38	0	1	0	1	0	0,04	0	0,015	15	16,5	0	0	0	0
37	14530	X1CrNiMoAlTi12-9-2	EN 10088	3,6	0	0,015	0	0,1	0	0,1	0	0,01	0	0,005	11,5	12,5	1,85	2,15	8,5	9,5
38	14313	X3CrNiMo13-4	EN 10088	2,2	0	0,05	0	0,7	0	1,5	0	0,04	0	0,015	12	14	3,5	4,5	0,3	0,7
39	10473	P355GH	EN 10028	0,67	0,1	0,22	0	0,6	1,1	1,7	0	0,025	0	0,01	0	0,3	0	0,08	0	0,3
40	10145	S275J2	EN 10025	0,67	0	0,18	0	0,5	0	1,5	0	0,025	0	0,025	0	0,29	0	0,11	0	0,42
41	14589	X5CrNiMoTi15-2	EN 10088	3,6	0	0,08	0	1	0	1	0	0,04	0	0,015	13,5	15,5	0,2	1,2	1	2,5
42	14736	X3CrAlTi18-2	EN 10088	2,2	0	0,04	0	1	0	1	0	0,04	0	0,015	17	18	0	0,1	0	0,5

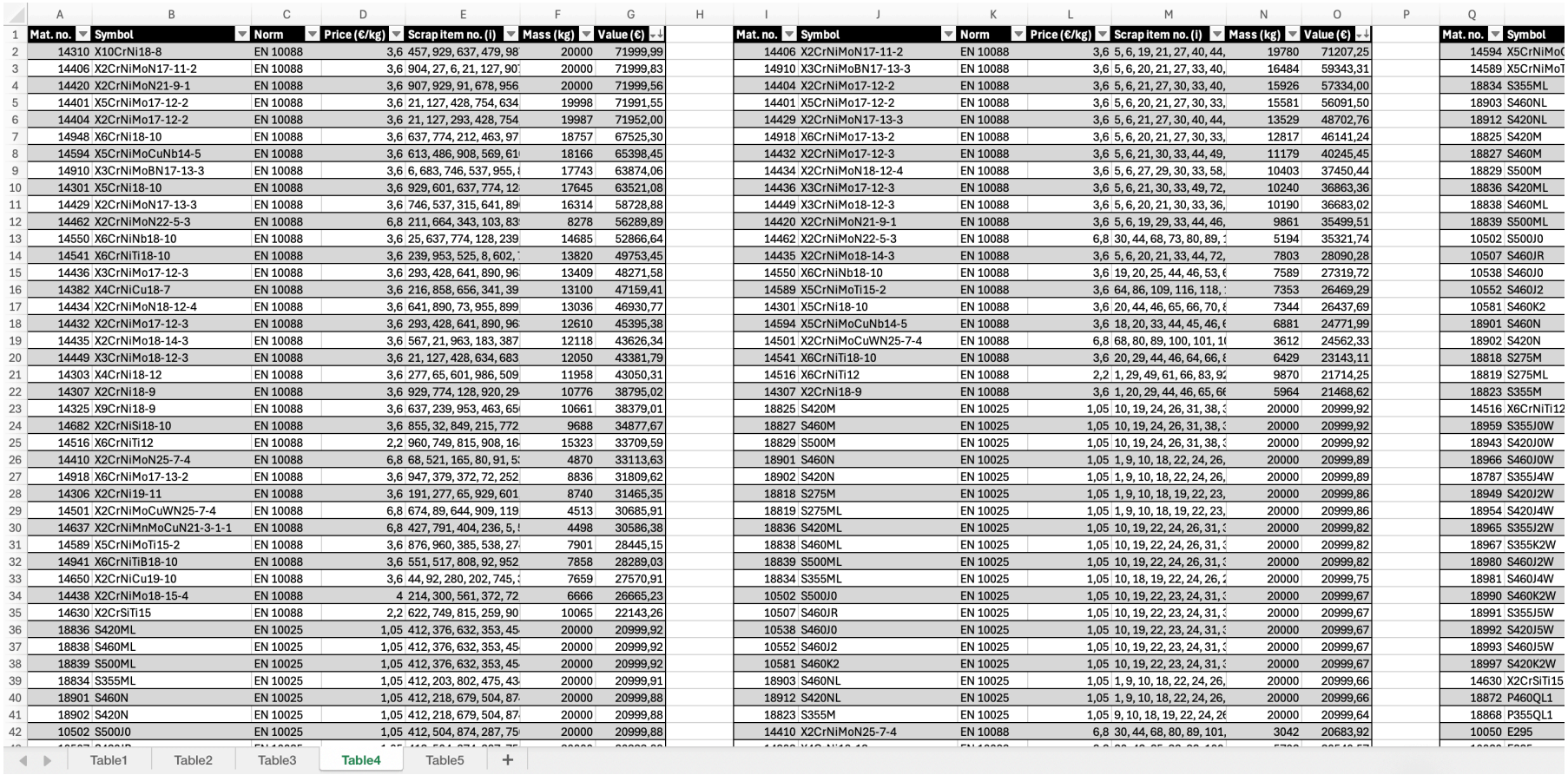
Table1 | Table2 | Table3 | Table4 | Table5 | +

Figure B.2: Screenshot of the candidate scrap table $\mathcal{T}_2$ in Microsoft Excel. This table was generated by sampling recipe rows with replacement from the input recipe table $\mathcal{T}_1$, thereby creating the candidate scrap items used in the data-generation procedure. The screenshot is included to illustrate the spreadsheet structure, only part of the full table is visible, as the table extends beyond the image frame.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Scrap item no. (i)	C %	Si %	Mn %	P %	S %	Cr %	Mo %	Ni %	N %	Cu %	Ti %	Nb %	V %	W %	Al %	Zr %	B %	Ce %	Co %	Mass (kg)
2	1	0,16	0,27	1,00	0,02	0,01	0,20	0,03	0,47	0,01	0,03	0,01	0,05	0,05	0,01	0,06	0,01	0,00	0,01	0,05	32
3	2	0,01	1,02	0,85	0,02	0,01	24,55	0,03	0,27	0,02	0,17	0,05	0,03	0,03	0,03	1,60	0,01	0,00	0,03	0,20	184
4	3	0,01	1,02	0,85	0,02	0,01	24,55	0,03	0,27	0,02	0,17	0,05	0,03	0,03	0,03	1,60	0,01	0,00	0,03	0,20	184
5	4	0,10	0,60	1,39	0,03	0,01	16,63	0,08	16,90	0,10	0,49	0,03	0,94	0,06	2,67	0,04	0,02	0,00	0,04	0,13	166
6	5	0,00	0,20	1,35	0,04	0,00	16,60	3,28	10,59	0,13	0,13	0,03	0,02	0,00	0,09	0,01	0,00	0,00	0,09	0,08	68
7	6	0,02	0,42	1,05	0,01	0,01	17,86	2,28	12,25	0,16	0,16	0,02	0,02	0,04	0,07	0,02	0,03	0,00	0,06	0,00	182
8	7	0,01	0,25	1,40	0,03	0,22	16,41	0,21	0,14	0,01	0,45	0,03	0,01	0,07	0,06	0,03	0,02	0,00	0,01	0,15	129
9	8	0,00	0,27	1,33	0,03	0,01	17,84	0,08	12,45	0,02	0,46	0,02	0,01	0,04	0,02	0,03	0,03	0,00	0,06	0,01	124
10	9	0,20	0,02	1,22	0,00	0,02	0,22	0,07	0,20	0,02	0,11	0,00	0,01	0,02	0,07	0,04	0,00	0,00	0,09	0,14	53
11	10	0,08	0,28	0,87	0,00	0,00	0,13	0,05	0,21	0,01	0,13	0,01	0,01	0,01	0,01	0,03	0,00	0,00	0,07	0,19	123
12	11	0,12	0,51	0,77	0,00	0,00	12,95	0,01	0,37	0,01	0,20	0,01	0,04	0,12	0,02	0,03	0,01	0,00	0,09	0,19	84
13	12	0,01	0,30	0,34	0,01	0,00	21,66	0,05	32,26	0,03	0,20	0,19	0,03	0,09	0,04	0,34	0,02	0,00	0,07	0,05	84
14	13	0,07	0,22	1,45	0,02	0,01	0,21	0,04	0,11	0,00	0,32	0,05	0,00	0,03	0,05	0,04	0,02	0,00	0,04	0,04	84
15	14	0,06	0,49	0,03	0,03	0,01	13,15	0,08	0,35	0,02	0,45	0,02	0,04	0,11	0,01	0,11	0,03	0,00	0,06	0,02	1
16	15	0,01	0,94	0,15	0,01	0,34	17,53	0,42	0,33	0,01	0,41	0,03	0,01	0,09	0,07	0,07	0,01	0,00	0,04	0,06	154
17	16	0,02	0,65	1,53	0,03	0,01	25,45	3,89	8,00	0,23	1,23	0,01	0,02	0,10	0,01	0,03	0,01	0,00	0,07	0,08	78
18	17	0,02	0,73	0,02	0,03	0,01	17,39	2,28	11,10	0,01	0,09	0,03	0,12	0,05	0,04	0,02	0,03	0,00	0,00	0,18	97
19	18	0,02	0,21	1,46	0,02	0,01	0,12	0,17	0,10	0,01	0,45	0,04	0,03	0,07	0,09	0,07	0,02	0,00	0,05	0,02	70
20	19	0,01	0,09	0,73	0,02	0,00	0,28	0,08	0,00	0,02	0,06	0,05	0,02	0,08	0,09	0,01	0,02	0,00	0,01	0,19	125
21	20	0,10	0,67	0,56	0,01	0,01	20,42	0,05	31,79	0,00	0,18	0,20	0,04	0,07	0,07	0,35	0,02	0,00	0,00	0,19	32
22	21	0,01	0,99	1,45	0,01	0,00	17,70	2,37	11,59	0,08	0,31	0,04	0,05	0,04	0,06	0,01	0,03	0,00	0,07	0,05	181
23	22	0,07	0,41	0,52	0,02	0,01	0,27	0,05	0,27	0,01	0,13	0,04	0,03	0,04	0,04	0,08	0,03	0,00	0,07	0,14	100
24	23	0,11	0,07	0,57	0,00	0,00	9,39	0,87	0,07	0,04	0,08	0,00	0,09	0,20	0,03	0,02	0,00	0,00	0,03	0,05	80
25	24	0,02	0,38	0,96	0,00	0,01	0,37	0,10	0,49	0,00	0,39	0,04	0,00	0,05	0,03	0,07	0,02	0,00	0,05	0,11	142
26	25	0,08	0,89	0,36	0,00	0,01	18,24	0,00	11,29	0,01	0,22	0,00	0,88	0,05	0,07	0,03	0,00	0,00	0,06	0,19	171
27	26	0,07	0,42	0,32	0,02	0,00	0,27	0,06	0,96	0,01	0,18	0,05	0,01	0,06	0,01	0,07	0,11	0,00	0,08	0,16	109
28	27	0,01	0,17	1,41	0,00	0,00	17,43	2,35	11,73	0,18	0,23	0,04	0,05	0,12	0,07	0,07	0,03	0,00	0,02	0,17	189
29	28	0,11	0,02	0,86	0,02	0,01	0,14	0,08	0,45	0,01	0,18	0,01	0,03	0,05	0,05	0,06	0,01	0,00	0,06	0,06	68
30	29	0,07	1,15	0,82	0,00	0,00	12,00	0,05	0,40	0,01	0,41	0,02	0,00	0,06	0,09	1,17	0,02	0,00	0,08	0,05	9
31	30	0,14	0,12	0,87	0,02	0,01	11,30	1,82	2,20	0,02	0,15	0,04	0,03	0,03	0,02	0,06	0,02	0,00	0,02	0,01	19
32	31	0,17	0,28	1,53	0,01	0,01	0,10	0,08	0,43	0,00	0,05	0,01	0,04	0,11	0,09	0,08	0,00	0,00	0,08	0,05	104
33	32	0,03	0,98	0,31	0,03	0,00	19,49	0,41	9,34	0,10	0,00	0,00	0,01	0,11	0,05	0,01	0,01	0,00	0,07	0,04	130
34	33	0,09	0,31	1,04	0,02	0,01	16,06	1,81	16,90	0,03	0,01	0,03	0,52	0,02	0,05	0,02	0,02	0,00	0,07	0,05	46
35	34	0,03	0,31	0,68	0,01	0,02	0,54	0,22	0,32	0,01	0,29	0,08	0,04	0,06	0,04	0,03	0,07	0,00	0,01	0,17	124
36	35	0,09	0,01	1,19	0,00	0,16	13,19	0,20	0,45	0,02	0,22	0,00	0,02	0,04	0,03	0,03	0,00	0,00	0,04	0,12	187
37	36	0,09	0,57	7,39	0,00	0,00	17,13	3,08	0,17	0,09	0,26	0,05	0,01	0,08	0,07	0,08	0,01	0,00	0,08	0,00	99
38	37	0,02	0,26	0,35	0,02	0,17	11,70	0,44	0,25	0,01	0,17	0,02	0,03	0,09	0,10	0,05	0,01	0,00	0,05	0,19	97
39	38	0,03	0,10	1,22	0,01	0,01	0,29	0,19	0,21	0,01	0,18	0,05	0,02	0,02	0,07	0,07	0,01	0,00	0,07	0,07	197
40	39	0,18	0,09	0,37	0,02	0,00	0,03	0,08	0,38	0,02	0,31	0,00	0,04	0,02	0,03	0,01	0,00	0,00	0,02	0,06	102
41	40	0,01	0,91	1,31	0,03	0,00	16,87	2,23	10,86	0,06	3,38	0,03	0,05	0,06	0,01	0,06	0,03	0,00	0,02	0,10	144
42	41	0,22	0,42	0,57	0,02	0,01	11,38	0,87	0,49	0,00	0,26	0,02	0,01	0,35	0,00	0,02	0,01	0,00	0,02	0,02	4

Table1 | Table2 | Table3 | Table4 | Table5 | +

Figure B.3: Screenshot of the generated scrap-item table $\mathcal{T}_3$ in Microsoft Excel. Each row represents a generated scrap item, including its elemental composition and assigned mass. This table forms the output of the scrap data-generation procedure described in Section 5.3 and is used as input for the sorting simulations. The screenshot is included to illustrate the spreadsheet structure, only part of the full table is visible, as the table extends beyond the image frame.



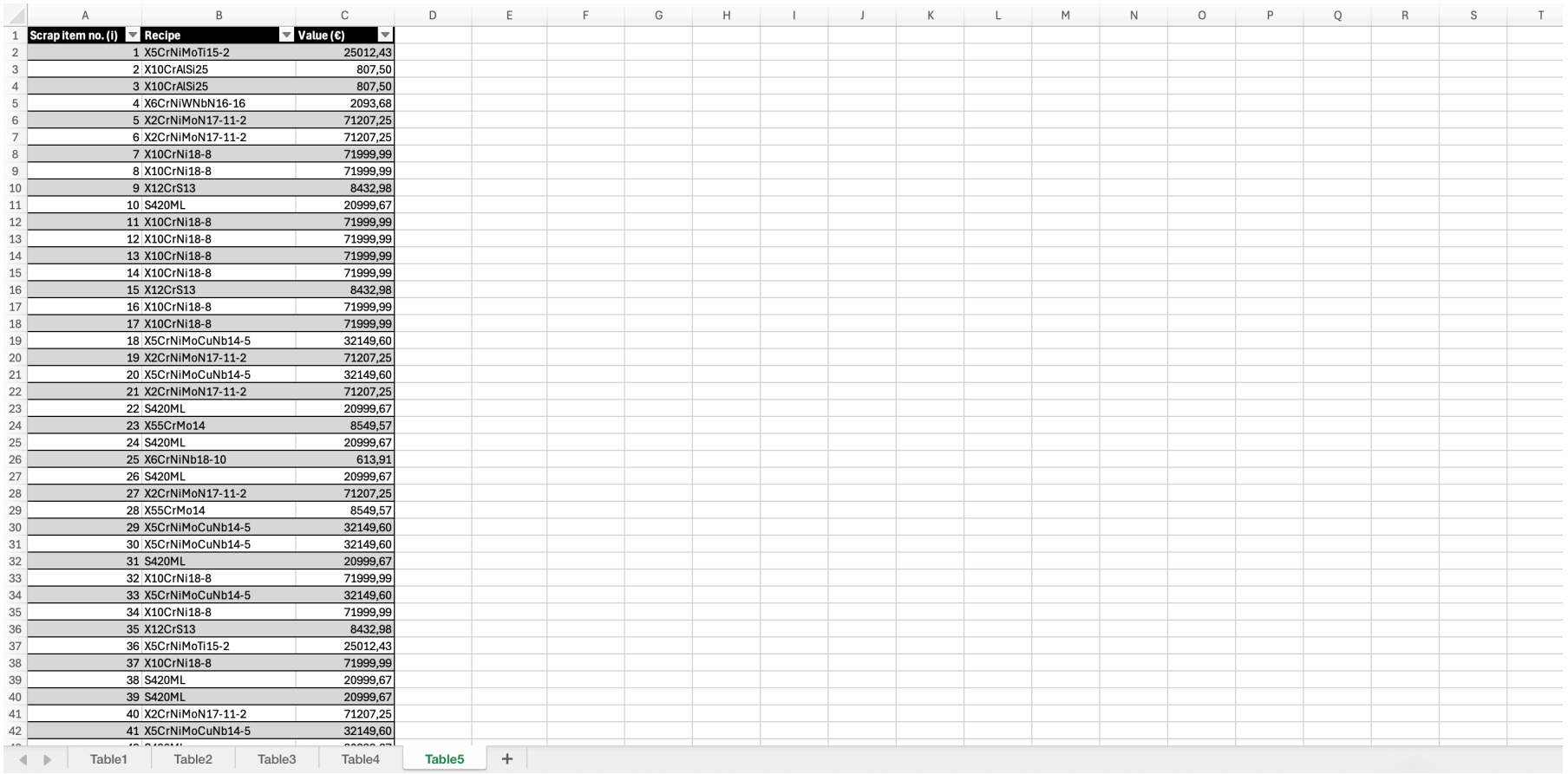
Mat. no.	Symbol	Norm	Price (€/kg)	Scrap item no. (i)	Mass (kg)	Value (€)
14310	X10CrNi18-8	EN 10088	3,6	457, 929, 637, 479, 98	20000	71999,99
14406	X2CrNiMoN17-11-2	EN 10088	3,6	904, 27, 6, 21, 127, 907	20000	71999,83
14420	X2CrNiMoN21-9-1	EN 10088	3,6	907, 929, 91, 678, 956	20000	71999,56
14401	X5CrNiMo17-12-2	EN 10088	3,6	21, 127, 428, 754, 634	19998	71991,55
14404	X2CrNiMo17-12-2	EN 10088	3,6	21, 127, 293, 428, 754	19987	71952,00
14948	X6CrNi18-10	EN 10088	3,6	637, 774, 212, 463, 97	18757	67525,30
14594	X5CrNiMoCuNb14-5	EN 10088	3,6	613, 486, 908, 569, 61	18166	65398,45
14910	X3CrNiMoBN17-13-3	EN 10088	3,6	6, 683, 746, 537, 955, 8	17743	63874,06
14301	X5CrNi18-10	EN 10088	3,6	929, 601, 637, 774, 12	17645	63521,08
14429	X2CrNiMoN17-13-3	EN 10088	3,6	746, 537, 315, 641, 89	16314	58728,88
14462	X2CrNiMoN22-5-3	EN 10088	6,8	211, 664, 343, 103, 83	8278	56289,89
14550	X6CrNiNb18-10	EN 10088	3,6	25, 637, 774, 128, 239	14685	52866,64
14541	X6CrNiTi18-10	EN 10088	3,6	239, 953, 525, 8, 602, 7	13820	49753,45
14436	X3CrNiMo17-12-3	EN 10088	3,6	293, 428, 641, 890, 96	13409	48271,58
14382	X4CrNiCu18-7	EN 10088	3,6	216, 858, 656, 341, 39	13100	47159,41
14434	X2CrNiMoN18-12-4	EN 10088	3,6	641, 890, 73, 955, 899	13036	46930,77
14432	X2CrNiMo17-12-3	EN 10088	3,6	293, 428, 641, 890, 96	12610	45395,38
14435	X2CrNiMo18-14-3	EN 10088	3,6	567, 21, 963, 183, 387	12118	43626,34
14449	X3CrNiMo18-12-3	EN 10088	3,6	21, 127, 428, 634, 683	12050	43381,79
14303	X4CrNi18-12	EN 10088	3,6	277, 65, 601, 986, 509	11958	43050,31
14307	X2CrNi18-9	EN 10088	3,6	929, 774, 128, 920, 29	10776	38795,02
14325	X9CrNi18-9	EN 10088	3,6	637, 239, 953, 463, 65	10661	38379,01
14682	X2CrNiSi18-10	EN 10088	3,6	855, 32, 849, 215, 772	9688	34877,67
14516	X6CrNiTi12	EN 10088	2,2	960, 749, 815, 908, 16	15323	33709,59
14410	X2CrNiMoN25-7-4	EN 10088	6,8	68, 521, 165, 80, 91, 5	4870	33113,63
14918	X6CrNiMo17-13-2	EN 10088	3,6	947, 379, 372, 72, 252	8836	31809,62
14306	X2CrNi19-11	EN 10088	3,6	191, 277, 65, 929, 601	8740	31465,35
14501	X2CrNiMoCuWN25-7-4	EN 10088	6,8	674, 89, 644, 909, 119	4513	30685,91
14637	X2CrNiMnMoCuN21-3-1-1	EN 10088	6,8	427, 791, 404, 236, 5, 8	4498	30586,38
14589	X5CrNiMoTi15-2	EN 10088	3,6	876, 960, 385, 538, 27	7901	28445,15
14941	X6CrNiTiB18-10	EN 10088	3,6	551, 517, 808, 92, 952	7858	28289,03
14650	X2CrNiCu19-10	EN 10088	3,6	44, 92, 280, 202, 745, 3	7659	27570,91
14438	X2CrNiMo18-15-4	EN 10088	4	214, 300, 561, 372, 72	6666	26665,23
14630	X2CrSiTi15	EN 10088	2,2	622, 749, 815, 259, 90	10065	22143,26
18836	S420ML	EN 10025	1,05	412, 376, 632, 353, 45	20000	20999,92
18838	S460ML	EN 10025	1,05	412, 376, 632, 353, 45	20000	20999,92
18839	S500ML	EN 10025	1,05	412, 376, 632, 353, 45	20000	20999,92
18834	S355ML	EN 10025	1,05	412, 203, 802, 475, 43	20000	20999,91
18901	S460N	EN 10025	1,05	412, 218, 679, 504, 87	20000	20999,88
18902	S420N	EN 10025	1,05	412, 218, 679, 504, 87	20000	20999,88
10502	S500J0	EN 10025	1,05	412, 504, 874, 287, 75	20000	20999,88

Mat. no.	Symbol	Norm	Price (€/kg)	Scrap item no. (i)	Mass (kg)	Value (€)
14406	X2CrNiMoN17-11-2	EN 10088	3,6	5, 6, 19, 21, 27, 40, 44,	19780	71207,25
14910	X3CrNiMoBN17-13-3	EN 10088	3,6	5, 6, 20, 21, 27, 33, 40,	16484	59343,31
14404	X2CrNiMo17-12-2	EN 10088	3,6	5, 6, 21, 27, 30, 33, 40,	15926	57334,00
14401	X5CrNiMo17-12-2	EN 10088	3,6	5, 6, 20, 21, 27, 30, 33,	15581	56091,50
14429	X2CrNiMoN17-13-3	EN 10088	3,6	5, 6, 21, 27, 30, 40, 44,	13529	48702,76
14918	X6CrNiMo17-13-2	EN 10088	3,6	5, 6, 20, 21, 27, 30, 33,	12817	46141,24
14432	X2CrNiMo17-12-3	EN 10088	3,6	5, 6, 21, 30, 33, 44, 49,	11179	40245,45
14434	X2CrNiMoN18-12-4	EN 10088	3,6	5, 6, 27, 29, 30, 33, 58,	10403	37450,44
14436	X3CrNiMo17-12-3	EN 10088	3,6	5, 6, 21, 30, 33, 49, 72,	10240	36863,36
14449	X3CrNiMo18-12-3	EN 10088	3,6	5, 6, 20, 21, 30, 33, 36,	10190	36683,02
14420	X2CrNiMoN21-9-1	EN 10088	3,6	5, 6, 19, 29, 33, 44, 46,	9861	35499,51
14462	X2CrNiMoN22-5-3	EN 10088	6,8	30, 44, 68, 73, 80, 89, 1	5194	35321,74
14435	X2CrNiMo18-14-3	EN 10088	3,6	5, 6, 20, 21, 33, 44, 72,	7803	28090,28
14550	X6CrNiNb18-10	EN 10088	3,6	19, 20, 25, 44, 46, 53, 6	7589	27319,72
14589	X5CrNiMoTi15-2	EN 10088	3,6	64, 86, 109, 116, 118, 1	7353	26469,29
14301	X5CrNi18-10	EN 10088	3,6	20, 44, 46, 65, 66, 70, 8	7344	26437,69
14594	X5CrNiMoCuNb14-5	EN 10088	3,6	18, 20, 33, 44, 45, 46, 6	6881	24771,99
14501	X2CrNiMoCuWN25-7-4	EN 10088	6,8	68, 80, 89, 100, 101, 10	3612	24562,33
14541	X6CrNiTi18-10	EN 10088	3,6	20, 29, 44, 46, 64, 66, 8	6429	23143,11
14516	X6CrNiTi12	EN 10088	2,2	1, 29, 49, 61, 66, 83, 92	9870	21714,25
14307	X2CrNi18-9	EN 10088	3,6	1, 20, 29, 44, 46, 65, 66	5964	21468,62
18825	S420M	EN 10025	1,05	10, 19, 24, 26, 31, 38, 3	20000	20999,92
18827	S460M	EN 10025	1,05	10, 19, 24, 26, 31, 38, 3	20000	20999,92
18829	S500M	EN 10025	1,05	10, 19, 24, 26, 31, 38, 3	20000	20999,92
18901	S460N	EN 10025	1,05	1, 9, 10, 18, 22, 24, 26,	20000	20999,89
18902	S420N	EN 10025	1,05	1, 9, 10, 18, 22, 24, 26,	20000	20999,89
18818	S275M	EN 10025	1,05	1, 9, 10, 18, 19, 22, 23,	20000	20999,86
18819	S275ML	EN 10025	1,05	1, 9, 10, 18, 19, 22, 23,	20000	20999,86
18836	S420ML	EN 10025	1,05	10, 19, 22, 24, 26, 31, 3	20000	20999,82
18838	S460ML	EN 10025	1,05	10, 19, 22, 24, 26, 31, 3	20000	20999,82
18839	S500ML	EN 10025	1,05	10, 19, 22, 24, 26, 31, 3	20000	20999,82
18834	S355ML	EN 10025	1,05	10, 18, 19, 22, 24, 26, 2	20000	20999,75
10502	S500J0	EN 10025	1,05	10, 19, 22, 23, 24, 31, 3	20000	20999,67
10507	S460JR	EN 10025	1,05	10, 19, 22, 23, 24, 31, 3	20000	20999,67
10538	S460J0	EN 10025	1,05	10, 19, 22, 23, 24, 31, 3	20000	20999,67
10552	S460J2	EN 10025	1,05	10, 19, 22, 23, 24, 31, 3	20000	20999,67
10581	S460K2	EN 10025	1,05	10, 19, 22, 23, 24, 31, 3	20000	20999,67
18903	S460NL	EN 10025	1,05	1, 9, 10, 18, 22, 24, 26,	20000	20999,66
18912	S420NL	EN 10025	1,05	1, 9, 10, 18, 22, 24, 26,	20000	20999,66
18823	S355M	EN 10025	1,05	9, 10, 18, 19, 22, 24, 26	20000	20999,64
14410	X2CrNiMoN25-7-4	EN 10088	6,8	30, 44, 68, 80, 89, 101,	3042	20683,92

Mat. no.	Symbol
14594	X5CrNiMo
14589	X5CrNiMoT
18834	S355ML
18903	S460NL
18912	S420NL
18825	S420M
18827	S460M
18829	S500M
18836	S420ML
18838	S460ML
18839	S500ML
10502	S500J0
10507	S460JR
10538	S460J0
10552	S460J2
10581	S460K2
18901	S460N
18902	S420N
18818	S275M
14516	X6CrNiTi12
18959	S355J0W
18943	S420J0W
18966	S460J0W
18787	S355J4W
18949	S420J2W
18954	S420J4W
18965	S355J2W
18967	S355K2W
18980	S460J2W
18981	S460J4W
18990	S460K2W
18991	S355J5W
18992	S420J5W
18993	S460J5W
18997	S420K2W
14630	X2CrSiTi15
18872	P460QL1
18868	P355QL1
10050	E295

Table1 Table2 Table3 Table4 Table5 +

Figure B.4: Screenshot of the candidate-heap result tables $\mathcal{T}_{4,t}$ in Microsoft Excel. Each table represents the candidate heaps generated during allocation round t out of T rounds and contains the selected recipe, assigned scrap-item numbers, total heap mass, and corresponding value. For each new allocation round, a new candidate-heap result table is placed to the right of the previous table, separated by one empty column. The screenshot is included to illustrate the spreadsheet structure, only part of the full sheet is visible, as the tables extend beyond the image frame.



Scrap item no. (i)	Recipe	Value (€)
1	X5CrNiMoTi15-2	25012,43
2	X10CrAlSi25	807,50
3	X10CrAlSi25	807,50
4	X6CrNiWNbN16-16	2093,68
5	X2CrNiMoN17-11-2	71207,25
6	X2CrNiMoN17-11-2	71207,25
7	X10CrNi18-8	71999,99
8	X10CrNi18-8	71999,99
9	X12CrS13	8432,98
10	S420ML	20999,67
11	X10CrNi18-8	71999,99
12	X10CrNi18-8	71999,99
13	X10CrNi18-8	71999,99
14	X10CrNi18-8	71999,99
15	X12CrS13	8432,98
16	X10CrNi18-8	71999,99
17	X10CrNi18-8	71999,99
18	X5CrNiMoCuNb14-5	32149,60
19	X2CrNiMoN17-11-2	71207,25
20	X5CrNiMoCuNb14-5	32149,60
21	X2CrNiMoN17-11-2	71207,25
22	S420ML	20999,67
23	X55CrMo14	8549,57
24	S420ML	20999,67
25	X6CrNiNb18-10	613,91
26	S420ML	20999,67
27	X2CrNiMoN17-11-2	71207,25
28	X55CrMo14	8549,57
29	X5CrNiMoCuNb14-5	32149,60
30	X5CrNiMoCuNb14-5	32149,60
31	S420ML	20999,67
32	X10CrNi18-8	71999,99
33	X5CrNiMoCuNb14-5	32149,60
34	X10CrNi18-8	71999,99
35	X12CrS13	8432,98
36	X5CrNiMoTi15-2	25012,43
37	X10CrNi18-8	71999,99
38	S420ML	20999,67
39	S420ML	20999,67
40	X2CrNiMoN17-11-2	71207,25
41	X5CrNiMoCuNb14-5	32149,60

Table1 Table2 Table3 Table4 Table5 +

Figure B.5: Screenshot of the final allocation and annotation table $\mathcal{T}_5$ in Microsoft Excel. Each row represents a generated scrap item and records the recipe assigned to that item in the final allocation, together with the corresponding allocation value. The screenshot is included to illustrate the spreadsheet structure, only part of the full table is visible, as the table continues beyond the image frame.

C

Results | Appendix

Additional figure references of Sections 6.6 and 6.7

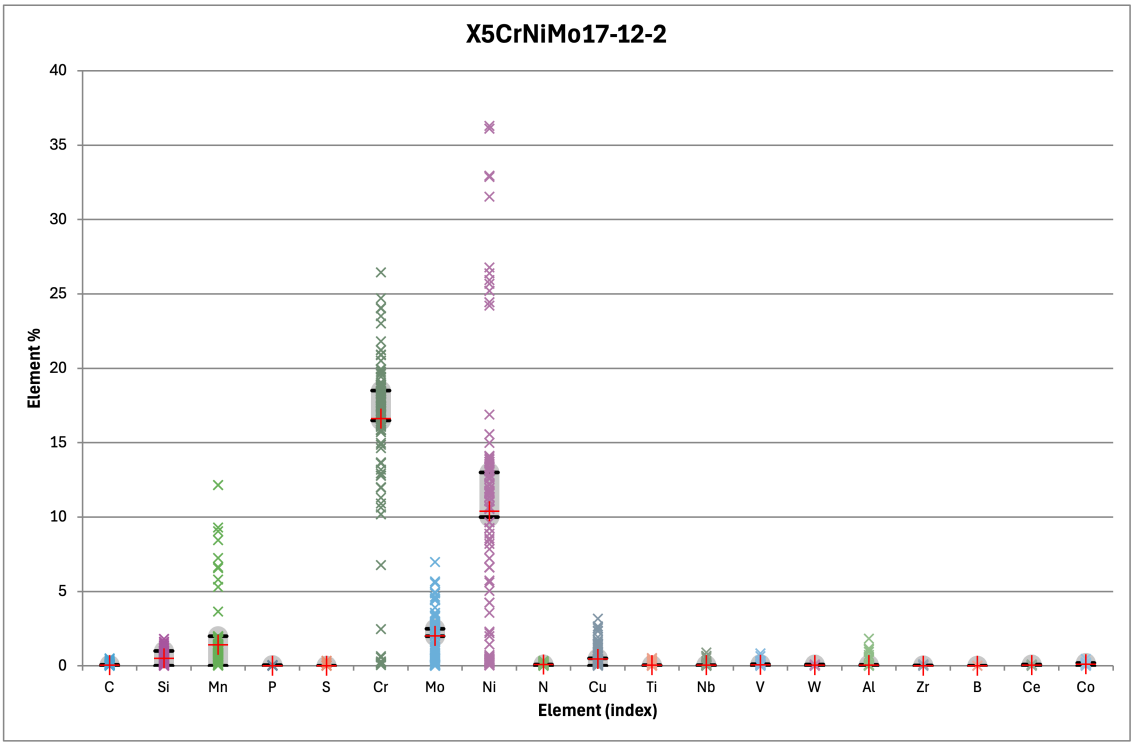


Figure C.1: Scrap composition per element for the third heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the X5CrNiMo17-12-2 steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

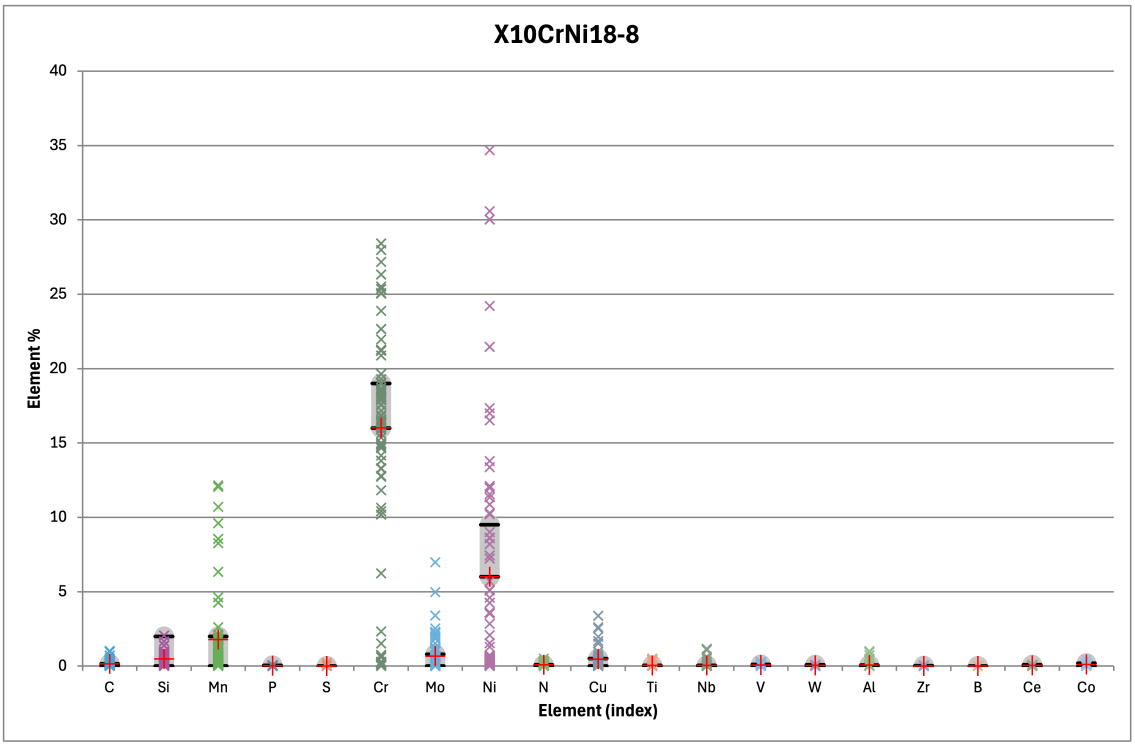


Figure C.2: Scrap composition per element for the fourth heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the X10CrNi18-8 steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

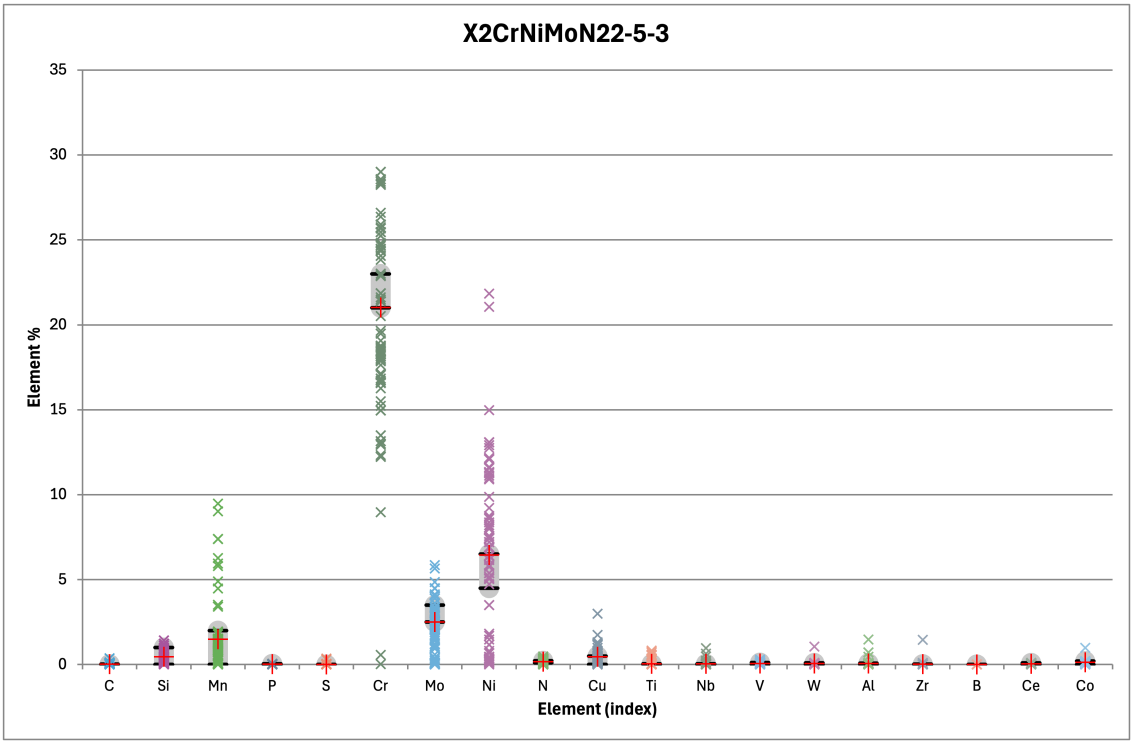


Figure C.3: Scrap composition per element for the fifth heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the X2CrNiMoN22-5-3 steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

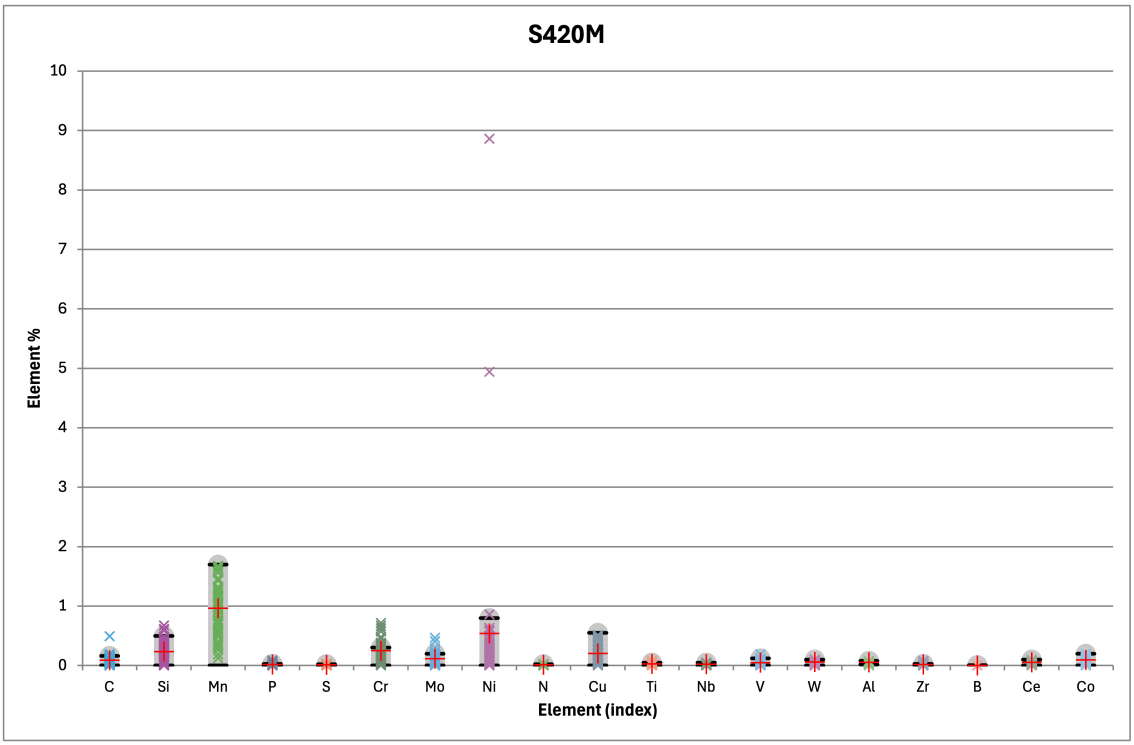


Figure C.4: Scrap composition per element for the sixth heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the S420M steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

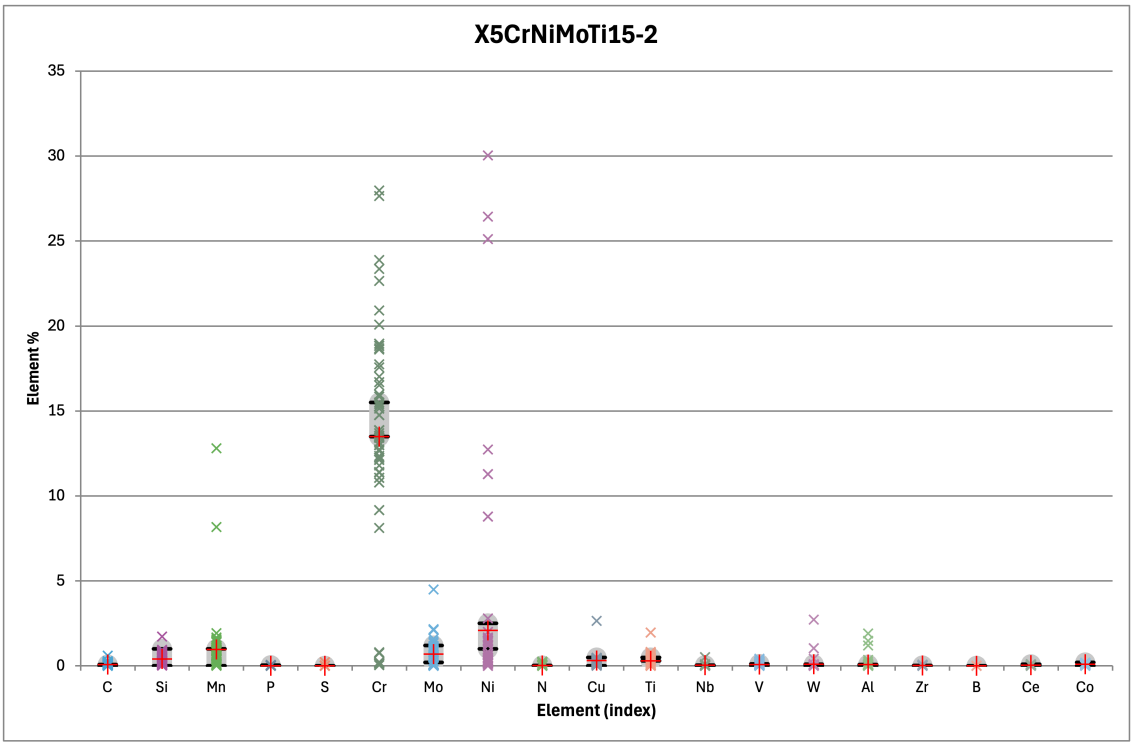


Figure C.5: Scrap composition per element for the seventh heap of the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, the black bars represent the allowable composition range of the X5CrNiMoTi15-2 steel recipe per element, and the red crosses represent the mass-weighted average scrap composition per element.

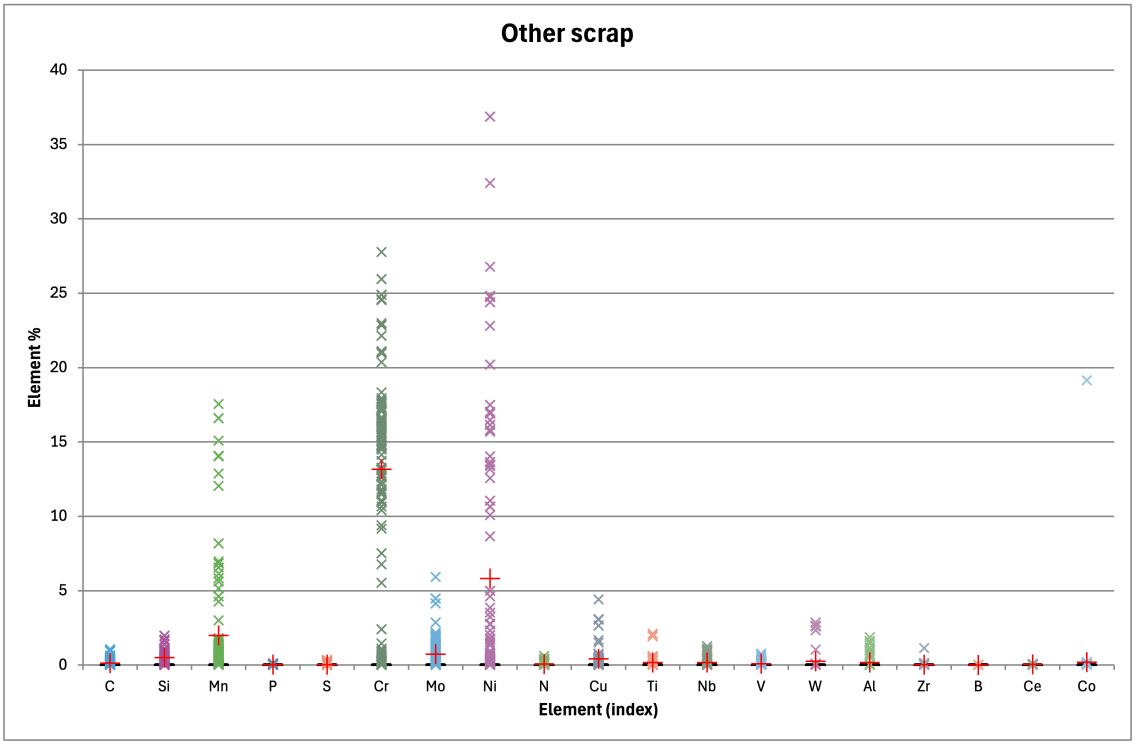


Figure C.6: Element-wise scrap composition of the remaining scrap items in the 7-heap configuration with a heap capacity of 12 000 kg per heap. Each point represents a scrap item, and the red crosses indicate the mass-weighted average composition.