Evaluating Decomposition Strategies to Enable Scalable Scheduling for a Real-World Multi-line Steel Scheduling Problem

Manal T. Adham
University College London
Computer Science
m.adham@ucl.ac.uk

Peter J. Bentley
University College London
Computer Science
peter.bentley@ucl.ac.uk

Diego Diaz
ArcelorMittal Global R&D
R&D Asturias
diego.diaz@arcelormittal.com

***Abstract*—Steel scheduling is recognised as one of the most difficult real-world scheduling problems. It is characterised by a wide range of operational constraints, variable dependencies and multiple objectives. This paper uses a divide and conquer method to reduce the combinatorial complexity of a real-world multi-line steel scheduling problem. The problem is first decomposed into sub-problems which are solved individually in parallel using parallel branch and bound, then sub-problems are combined to form a solution to the original problem. Three decomposition strategies are compared, specifically: a manual heuristic domain knowledge (DOM) intensive strategy, K-means++ (KM) clustering and Self-organising maps (SOM). Experimental results show that using SOM for decomposition is a promising approach. This paper demonstrates that despite being a highly complex and constrained problem, it is possible to use divide and conquer to achieve potentially good scalability characteristics without significant detriment to the solution quality.**

I. Introduction

We live in a globalised, hyper-connected, massively interdependent world. A significant challenge facing all countries today is Industry 4.0, the next digital industrial revolution [1]. The world around us is becoming more complex and a high demand for productivity gains to logistic problems persists [2]. Most scheduling problems found in industrial environments can be regarded as large-scale complex combinatorial optimisation problems (COP) classed as NP-Hard [3]. To date, no polynomial time complexity algorithm is known to solve COPs optimally.

A major difficulty in solving large-scale COPs is the computational expense involved. COPs are characterised by a rugged fitness landscape that contains a combinatorial number of local optima; this imposes high computational requirements on the underlying implementation platform. The complexity of COPs is heightened in real-world problems as a wide range of operational constraints often need to be taken into account, they are subjected to dynamic change over space and time, companies have limited computational resources, are deadline-sensitive, constraints are affected by uncertainty, there are multiple objective and constraint functions which can be expensive to evaluate and may have numeric or experimental noise.

The performance of many optimisation algorithms deteriorates [4] as the problem size increases. This can be attributed to an increase in the dimensionality of the search space (see: *the curse of dimensionality* [5]). Each schedule in a steel manufacturing plant has a respective cost associated with it, meaning the wrong decision can be very damaging if it has a substantial financial impact [6]. This necessitates identifying efficient optimisation strategies that explore promising regions of the search space within a feasible time-frame.

This paper solves a large-scale real-world multi-line steel scheduling problem. A natural approach (and the method used within natural ecosystems) is to tackle large-scale problems and reduce the combinatorial explosion is through a divide-and-conquer approach. This paper places emphasis on using decomposition (divide-and-conquer) to explore the underlying structure of the corresponding optimisation problem. The key idea is to reduce the complexity of the original problem by decomposing the problem of size n into b sub-problems of size $\frac{n}{b}$, solving their respective sub-problems then developing strategies to link partial solutions. In a realistic setting this size may vary between different sub-problems. This process also adds some linear complexity; however the benefit of scaling outweighs this cost.

The idea of decomposition (divide-and-conquer) appears in early work on large-scale linear programs in the 1960s [7]. The idea is to ensure the relationship between any two modules at the same level of abstraction is as weak as possible in order to create meaningful partitions. Decomposition allows the use of a parallel or distributed approach so that the computational load can be spread over a network, thereby offering scalability, since size and time requirements do not grow as significantly with an increase in the problem size [8]. Moreover this permits greater scope for flexibility as specialised methods can be applied to different sub-problems and solution building blocks can be re-used.

Finding sub-problems is a computationally difficult NP-hard task [9], therefore it is infeasible to apply a precise analytic algorithm. Despite this hardness there have been several methods proposed with varying success levels [10]. This

paper focuses on three strategies: a manual heuristic domain knowledge intensive (DOM) strategy, K-means++ (KM) and Self-organising maps (SOM).

This paper is organised as follows: The next section surveys related work on steel scheduling problems and solutions proposed. Section 3 gives details on the multi-lined steel scheduling problem considered. Section 4 describes the proposed approach. Section 5 provides details on the experiments and analyses results. The final section discusses findings and draws conclusions.

II. Related Work

Using a divide-and-conquer approach has successfully been applied to scheduling and planning. Research demonstrates this approach gives potential for better scalability [11, 12]. Commonly used decomposition algorithms include clustering [13], dual decomposition [14], column generation [15] as well as Dantzing Wolfe and Benders decomposition [16].

There are three common ways for tackling COPs like the multi-lined steel scheduling problem:

1) Commercial solvers (e.g. CPLEX [17]): often apply Mixed Integer Linear Program (MILP) and branch and bound to solve COPs within a specified tolerance.
2) Intelligent algorithms: includes heuristic algorithms that allow finding a feasible solution within a reasonable time-frame. This includes Simulated Annealing [18], as well as nature inspired algorithms like the Genetic Algorithm [19].
3) Constraint programming (CP): uses a mathematical model which embeds constraints and only searches the feasible region of candidate solutions [20].

Solutions are not limited to these categories; the current literature favours hybrid approaches that use a combination of these techniques [6, 22, 23, 21].

Harijunkoski and Grossmann [23] consider solving scheduling problems that involve cost minimisation, due dates and sequence independent setup times for a test problem. They decompose a multistage steel scheduling problem into an assignment problem and a sequencing problem. A hybrid MILP and CP model is used, where MILP is used for the assignment problem and MILP or CP is used for the sequencing problem. Numerical results show the approach can be applied in an industrial scale.

Tang *et al.* [24] tackle coil sequencing problem to ensure the switching cost between consecutive coils is minimised whilst satisfying constraints. The problem is divided into subproblems, each corresponds to a sequence of coils processed sequentially. The coils in each turn are then grouped based on width. Then a two-phase dynamic programming (DP) algorithm is proposed. In the first phase boundary coils (first and last coils) are identified. In phase two, a DP is used to form a complete sequence whilst taking into account boundary coils.

Fernandez *et al.* [25] schedule the galvanisation of steel using Ant Colony Optimisation. Transition costs for each combination of two nodes are calculated. These costs are then classified into zero, finite (soft constraints violated) and infinite (hard constraints violated). Transition costs are then probabilistically selected to construct a solution.

From our literature survey, we found that a multi-line steel scheduling problem for the production of galvanised steel has seldom been tackled. The scheduling problem in this paper uses real-world data and each line has a number of soft and hard constraints.

III. The Multi-line Steel Scheduling Problem

The galvanisation of steel involves coating it with a zinc layer to protect against air and moisture. The production of galvanised steel involves processing steel coils sequentially through different lines and yards. This work aims to reduce the cost of scheduling the following 4 lines: Pickling, Tandem Mill, Hot Dip Galvanising 1 and 2. Steel coils sequentially flow through these different lines - a topology is presented in Figure.1.

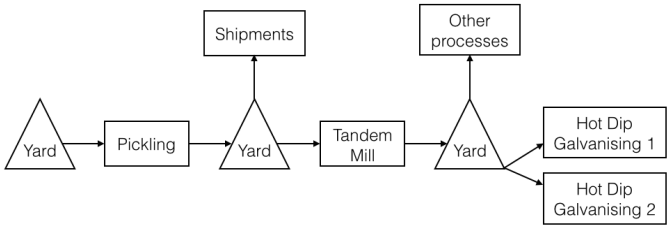


Fig. 1. Topology of the multiple lines considered.

Coils go through different lines/yards in a tree structure, whereby no coil goes through all stages. Each coil corresponds to a production order, which needs to be processed according to an order specification. They have different grades that can complement or hinder each other during the production process. They are also welded together and processed sequentially. Coils in a sequence must have similar properties, as the machine settings gradually change over time. All coils are proceed through the same lines, despite the differences in the required characteristics of the final product.

Coils are distinguished by their width, thickness, length, chemical composition and grade. Each grade has a given production recipe with strict specifications of temperature, speed, chemistry, and processing times at different production stages. It is important to ensure steel grade constraints overlap in order to avoid loss of material. The key to increased productivity and quality is to ensure that width profiles in a sequence are as smooth as possible. In this continuous process rollers are worn by processing coils. Processing narrower coils before wider coils causes edge marks to transfer to wider coils, which in effect degrades the quality of wider coils.

The large number of constraints related to the geometry and chemistry of the coils make it very difficult to obtain feasible solutions. Due to the NP-hard nature of this problem, it is more feasible to obtain good approximate (near) optimal solutions. Modularity allows the flexibility to adapt to changes in the environment e.g. when a new coil enters, only the module

relevant will be modified not the whole solution. Modules are groups of coils that have more connection to one another then otherwise expected if coils were randomly sampled. Modularity can be observed in ecosystems where different regions correspond to different species which facilitate specific functions [26].

Variations of this problem appear in almost any application domain. The original motivating application comes from industrial engineering which are a result of finite number of factory machines. Despite significant advances there are still major challenges and questions that remain unresolved.

IV. Proposed Approach

The goal is to schedule coils whilst ensuring steel quality is maximised and cost as well as lead time minimised. The cost takes into consideration material loss and line breakages, it is possible to lose meters of strip as a result of quality or a breakage that halts the facility for hours. Switching costs occur between coils in a sequence and the product quality is affected if the characteristics of a coil are different to the coil immediately after it. Essentially there are two costs that need to be minimised: the transition cost is the switching cost incurred to move between any two coils, and the sequencing cost is a function of the whole sequence [25].

Due to the confidential nature of this problem, a complete mathematical formulation of the problem cannot be provided. The proposed approach leverages decomposition to identify the underlying structure of the problem and facilitate solving large-scale real-world combinatorial optimisation problems (COP) in a scalable manner. The decomposition strategy is presented in Figure. 2.

The following steps are involved in the proposed approach:

1) Set the decomposition algorithm on a set of coils, to separate them into groups (sub-problems) with similar properties e.g. width, thickness, temperature and grade.
2) Run COP solver (branch and bound) on sub-problems independently and in parallel. Each sub-problem corresponds to a sequence of coils continuously processed in a schedule.
3) Combine sub-problems based on hard constraint costs. Major costs are a function of width, thickness and thermal changes.

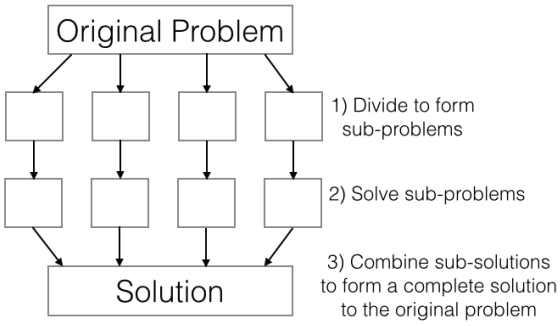


Fig. 2. Decomposition Strategy

A. Decomposition

Multi-stage optimisation scheduling problems can naturally be decomposed into assignment and sequencing subproblems [27]. This paper evaluates three strategies for problem decomposition:

1) Domain knowledge strategy (DOM) - Converts domain experience into a hierarchical set of rules used to group similar coils. For example, one rule is that coils with grade A cannot be followed by coils of grade B.
2) K-Means++ (KM) clusters [29, 28] - K-means++ was used to help improve centroid initialisation; experimental results using real-world datasets in [28] show a substantial improvement. The number of clusters k is first chosen using trial and error. Coils are then assigned to clusters based on their spatial proximity to the centroid. This distance is adapted to the different lines and allows coils with similar features to be grouped together. Coil features including width, thickness, thermal cycle and strength are considered.
3) Self-organising map (SOM) [30] - An unsupervised learning algorithm that allows the mapping from high dimensional space to a 2-dimensional space, whilst preserving the topological structure. A SOM model where the input is connected to every cell in the map is presented in Figure. 3 SOM is an artificial neural network that works in two phases, training and mapping. In the training phase, competitive learning is used by neurons to learn from a sample, thereby allowing a topological ordering of the map. In the mapping phase, input vectors are classified. There are three important variables that must be adjusted to suit the problem size: the height and width of the 2-dimensional map and the neighborhood radius. Specific coil characteristics including width, thickness and thermal cycle are used as features in the SOM, slightly different characteristics are used for the different lines. An intriguing feature of SOMs is that the number of clusters does not need to be specified: coils with similar features are grouped together.

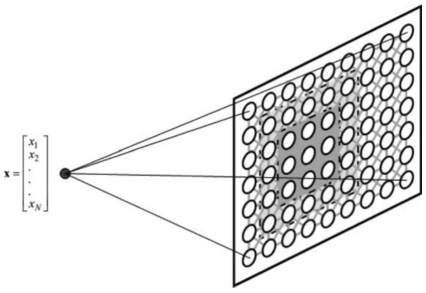


Fig. 3. Self-organising map model [31]

B. COP Solver

In order to evaluate the effectiveness of the different decomposition strategies, in this work a standard parallel branch

TABLE I
EXAMPLE COIL DATA (OBFUSCATED FOR DATA PROTECTION)

ID	Width	Thickness	Length	Weight	Grade
COIL1	600	4.4	500	2000	A
COIL2	100	4.3	700	3000	B
COIL3	200	5.1	800	3000	C
COIL4	200	5.1	700	3000	A

and bound (BB) [32] is used to solve all sub-problems. The algorithm explores branches of a tree, each representing a set of solutions, before enumerating through a branch. The upper and lower bounds are estimated using transition costs, then a decision is made on whether to explore a particular branch or not. BB is an enumerative algorithm that includes a pruning criteria to help navigate through the search space in order to avoid performing an exhaustive search. It is important to sort coils based on width prior to running BB. BB can be substituted with other approaches like Genetic Algorithm [33] and Simulated annealing [18]. Alternatively a combination of different approaches can also be used depending on the nature of the sub-problem.

V. EXPERIMENTS

Three test cases were executed across the four lines, pickling, tandem mill, hot dip galvanising 1 and 2. Experiments were carried out on real world data: the number of coils for each test case ranges between 60-90.

The three test cases cover different environments for the scheduler. The first test case is a typical one-day batch taken from production history, which ensures that there is a relatively smooth solution. The second test case is larger by 50%, and is also taken from production history. The third case is the same size as the first batch, but it is designed to mimic the batches that could be expected when working with lower stock levels, it is harder to schedule as there are fewer options for finding compatible coils.

In order to ensure reliability 30 independent runs were carried out using three different test cases on all four lines. The experiments were performed to evaluate the three different decomposition methods: Domain-specific Manual Heuristics (DOM), K-means++ (KM) and Self-Organising Maps (SOM). The values for k clusters, and SOM map width, height and neighbourhoods were set using running trial and error experiments and the best combination of values found was then used for the following experiments.

All experiments were performed using a OS X machine with 2.8 GHz Intel Core i7 and 16GB memory. The different sub-problems were executed in parallel using threads.

A. Solution Quality

The goal of sequencing is to maximise productivity and solution quality whilst minising cost and lead time. The schedule quality is determined with respect to three costs:

1) Cost functions: costs associated with placing two subsequent coils in a sequence.
2) Soft constraints: it is preferable not to violate soft constraints as they impose an additional cost to the sequence.
3) Hard constraints: which should not be violated but at certain situations it is impossible to produce a schedule without violating some hard constraints. Hard constraints impose a very high cost to the sequence in comparison with the soft constraints.

TABLE II
THE NUMBER OF COST FUNCTIONS, SOFT CONSTRAINTS AND HARD CONSTRAINTS FOR EACH LINE.

	Cost Functions	Soft Constraints	Hard Constraints
Pickling Line (PKL)	3	1	5
Tandem Mill Line (TDM)	3	0	3
Hot Dip Galvanizing 1 Line (HDG2)	5	0	6
Hot Dip Galvanizing 2 Line (HDG1)	4	1	5

TABLE III
WEIGHTS APPLIED TO AVOID BREAKING CONSTRAINTS.

Weight for Soft Constraints	1000
Wight for Hard Constraints	10000000

B. Results and Analysis

Three different test cases were performed using the three decomposition strategies (DOM, KM and SOM) as well as just using a parallel branch and bound (BB) across the four different lines (PKL, TDM, HDG1 and HDG2). The overall solution sequence cost is a weighted sum of the cost function, soft constraints and hard constraints, where the hard constraints have a considerably high weight to try to avoid them. The number cost functions, soft constraints and hard constraints is presented in Figure 7.

The performance of all three decomposition strategies and a parallel branch and bound (BB) is presented for all four lines (PKL, TDM, HDG1 and HDG2) in Figures 4, 5 and 6, in terms of cost functions, soft and hard constraints. The overall results of all three test cases for solutions across all lines is presented in Figure 7. The standard deviation measures are also presented in all figures using error bars to show the dispersion of the results.

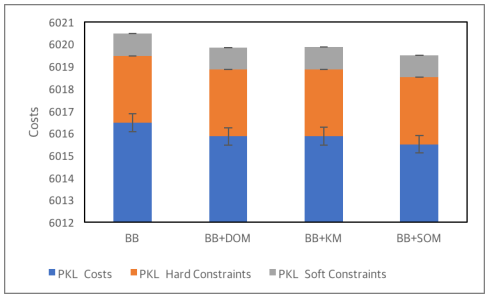
Test case 1 results are presented in Figure 4. For the PKL line it is visible that all approaches have a similar level of performance. For the TDM line it is visible that SOM performs considerably better than all other approaches, whereas KM performs considerably badly as a high number of hard constraints are violated. For HDG1, SOM performs a little better than DOM and BB, whilst KM does not seem to be performing well. For HDG2, BB is performing the best, this

is followed by KM, SOM then DOM. The error bars which represent the standard deviation are very narrow showing the reliability and confidence in the results is high. The error bars which represent the standard deviation are very narrow for PKL and TDM, this shows very high reliability and confidence in the results. These bars are slightly wider for the galvanizing lines, showing there is a little more variability in the results.

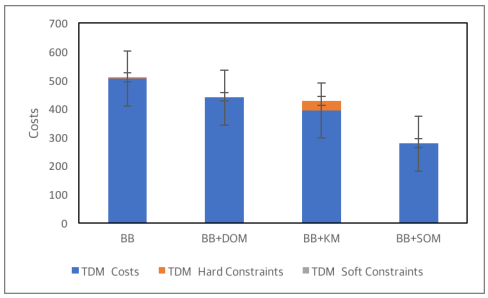
Test case 2 results are presented in Figure 5. For the PKL line, KM seems to have switched roles and is now performing much better than all the other methods, this is followed by BB, DOM and finally SOM. For the TDM line, all lines have similar performance. For the HDG1 line, BB is performing the best, this is followed by KM, DOM and then SOM. For HDG2, the best performing algorithms in order are BB, KM, DOM then SOM. The error bars represent the standard deviation are very narrow throughout all the lines, this shows the reliability and confidence in the results is high and that variability is low. For all lines the test case BB is working considerably well, this is followed by KM/DOM and then SOM.

Test case 3 results are presented in Figure 6. For the PKL line, the best performing algorithm is BB, then KM, SOM and finally DOM. For the TDM line, the best performing algorithm is SOM, all others have similar performance. For HDG1, BB is the best forming, this is followed by SOM, KM and finally DOM. For HDG2, all algorithms have very similar results. There error bars show there is some variability in the results, which is very low for the TDM and HDG2. This test shows that BB and SOM are performing considerably well compared to KM and DOM.

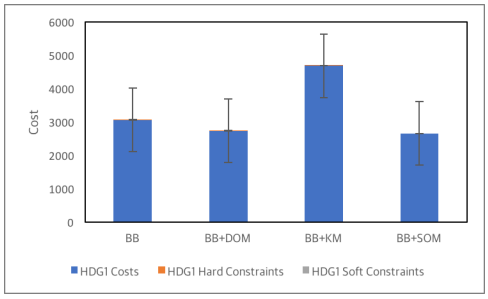
Average results across all 3 test cases is presented in Figure 7. These figures represent a clearer view of the actual performance of different cost functions as the weights for the different costs are considered and the values are averages of all test cases. For PKL, it is visible that SOM, KM and BB are all performing equally. For TDM, SOM is performing better than all the other approaches including BB, it is followed closely by DOM. For HDG1, BB is performing the best, followed by KM then closely by SOM, For HDG2, BB is performing the best, this is followed by SOM. The error bars are very narrow for TDM and HDG2 showing high confidence and reliability with low variability in the results: these are slightly wider for PKL where all the approaches seem to be performing very similarly, the most variability is visible in HDG1, where its clear that domain heuristic approach seems have the highest variability.



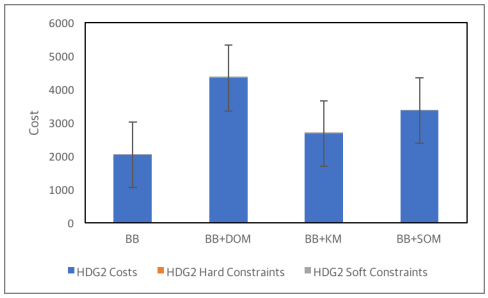
(a) Performance comparison for PKL



(b) Performance comparison for TDM



(c) Performance comparison for the HDG1



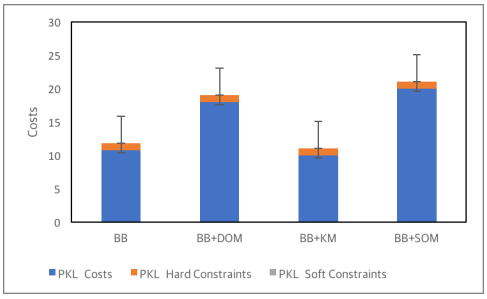
(d) Performance comparison for the HDG2

Fig. 4. Performance comparison for Test Case 1

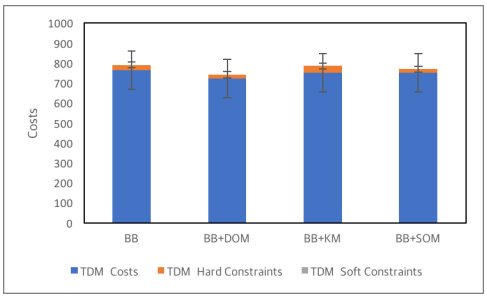
VI. Conclusion

The multi-lined steel scheduling problem is computationally difficult and requires considerable expertise in optimisation and the application domain. This paper evaluated solving the problem using a parallel branch and bound and three different decomposition strategies (DOM, KM and SOM). Results show the potential significance of using SOM as a decomposition strategy, SOM proved to be competitive with just using BB. Using SOM allows decomposing the problem without domain knowledge: problem features are selected and decomposition is performed on these features.

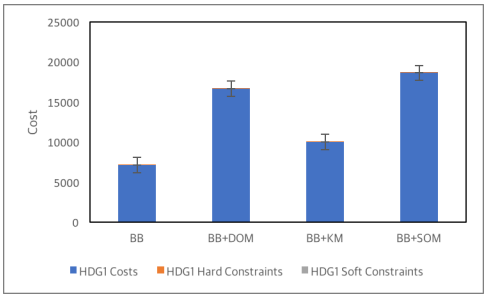
These results carry significance as SOMs are not commonly viewed as decomposition algorithms that can be applied to real-world highly constrained problems. It is important to note that no improvements were performed on the solutions after combining sub-solutions obtained. However, it is feasible to perform further improvements on the whole solution. This is of significance as decomposition methods are usually heavily dependent on the underlying problem. A simple SOM has



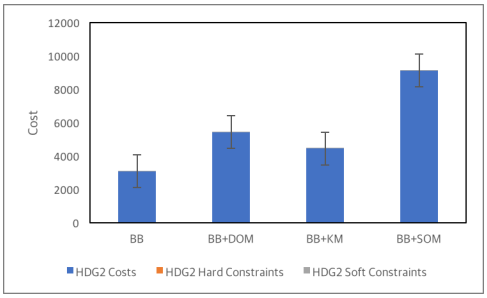
(a) Performance comparison for PKL



(b) Performance comparison for TDM

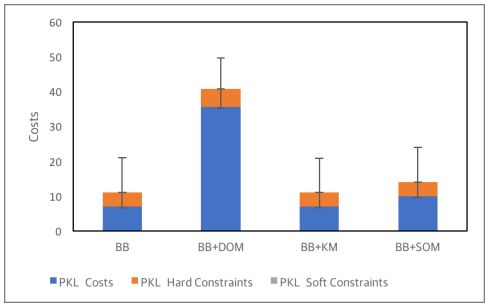


(c) Performance comparison for the HDG1

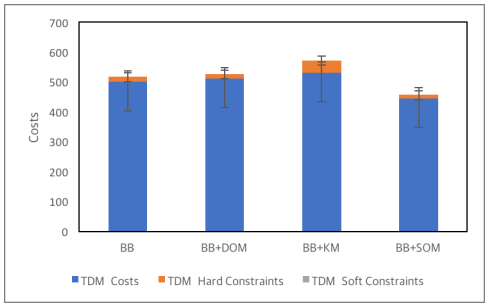


(d) Performance comparison for the HDG2

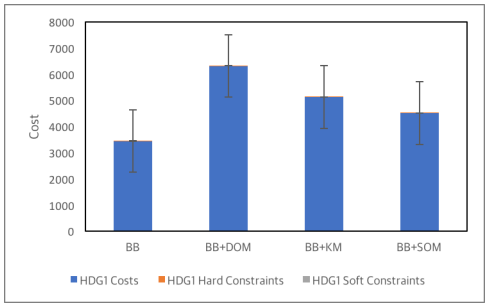
Fig. 5. Performance comparison for Test Case 2



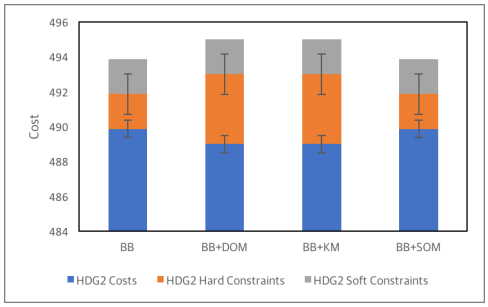
(a) Performance comparison for PKL



(b) Performance comparison for TDM



(c) Performance comparison for the HDG1



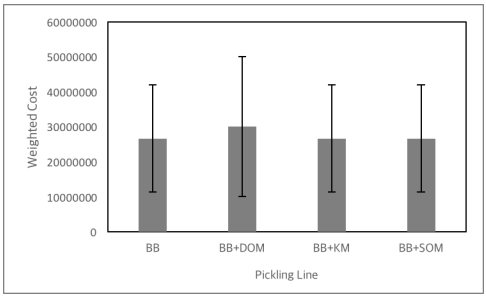
(d) Performance comparison for the HDG2

Fig. 6. Performance comparison for Test Case 3

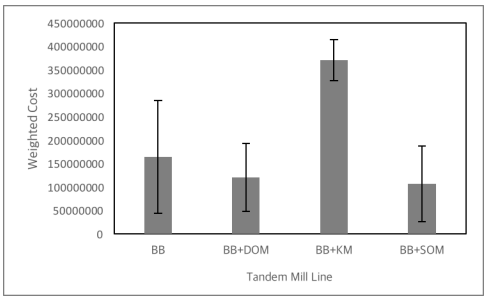
shown surprisingly good results, further advances on the SOM may provide better results.

SOM demonstrates potentially good scalability characteristics that can be applied to highly complex and constrained problems without significant detriment to the solution quality, the solutions found using this approach are competitive with BB and can potentially scale to larger problems. Furthermore, this modular approach provides increased flexibility, allowing the use of different strategies for different sub-problems.

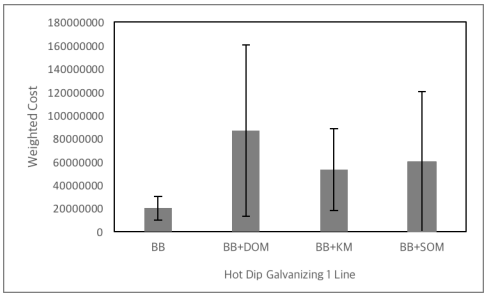
Data decomposition methods are not limited to SOM and K-means++, different levels of clustering can be used to expose different dimensions of the data. For example, density-based clustering can be used to avoid the formation of highly packed clusters as well as dual and column based decomposition strategies. Dividing a problem into sub-problems is an NP-hard task in itself. For this reason it is important to evaluate and compare different decomposition strategies, especially when the problem is complex and highly constrained.



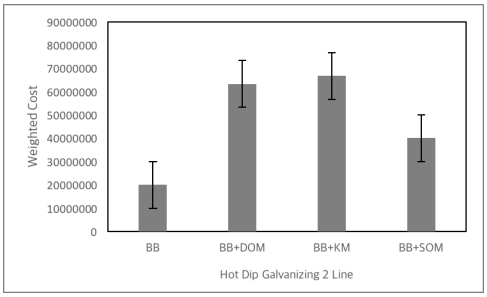
(a) Weighted cost for PKL



(b) Weighted cost for TDM



(c) Weighted cost for the HDG1



(d) Weighted cost for the HDG2

Fig. 7. Weighted cost averages for Test Cases 1, 2 and 3

This proposed approach does not guarantee globally optimal solutions, this is not possible as it is an NP-Hard problem, but rather it allows a solution that has potential to upscale to large-scale real world problems. This approach is generic as well as flexible and can be adapted to solve real-world COPs in other contexts.

ACKNOWLEDGMENT

The authors would like to thank ArcelorMittal for their help and cooperation in this project.

REFERENCES

[1] Mario Hermann, Tobias Pentek, and Boris Otto. "Design principles for industrie 4.0 scenarios". In: *System Sciences (HICSS), 2016 49th Hawaii International Conference on*. IEEE. 2016, pp. 3928–3937.

[2] Mirka Kans and Diego Galar. "The Impact of Maintenance 4.0 and Big Data Analytics within Strategic Asset Management". In: *6th International Conference on Maintenance Performance Measurement and Management, 28 November 2016, Luleå, Sweden*. Luleå University of Technology. 2017, pp. 96–103.

[3] William J Cook et al. *Combinatorial optimization*. Vol. 605. Springer, 1998.

[4] Jürg Nievergelt. "Exhaustive search, combinatorial optimization and enumeration: Exploring the potential of raw computing power". In: *International Conference on Current Trends in Theory and Practice of Computer Science*. Springer. 2000, pp. 18–35.

[5] Richard Bellman. "Dynamic programming and Lagrange multipliers". In: *Proceedings of the National Academy of Sciences* 42.10 (1956), pp. 767–769.

[6] Karla L Hoffman. "Combinatorial optimization: Current successes and directions for the future". In: *Journal of computational and applied mathematics* 124.1 (2000), pp. 341–360.

[7] George B Dantzig and Philip Wolfe. "Decomposition principle for linear programs". In: *Operations research* 8.1 (1960), pp. 101–111.

[8] Rolf Riesen, Ron Brightwell, and Arthur B Maccabe. "Differences between distributed and parallel systems". In: *SAND98-2221, Unlimited Release, Printed October* (1998).

[9] William J Welch. "Algorithmic complexity: three NP-hard problems in computational statistics". In: *Journal of Statistical Computation and Simulation* 15.1 (1982), pp. 17–25.

[10] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. "Data clustering: a review". In: *ACM computing surveys (CSUR)* 31.3 (1999), pp. 264–323.

[11] Samuel A Mulder and Donald C Wunsch. "Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks". In: *Neural Networks* 16.5 (2003), pp. 827–832.

[12] Yi Mei, Ke Tang, and Xin Yao. "Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem". In: *IEEE Transactions on Evolutionary Computation* 15.2 (2011), pp. 151–165.

[13] Mohammad Amin Adibi and Jamal Shahrabi. "A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem". In: *The International Journal of Advanced Manufacturing Technology* 70.9-12 (2014), pp. 1955–1961.

[14] Paul Swoboda, Jan Kuske, and Bogdan Savchynskyy. "A dual ascent framework for Lagrangean decompo-

sition of combinatorial problems". In: *arXiv preprint arXiv:1612.05460* (2016).

[15] Jian-Ya Ding et al. "Parallel machine scheduling under time-of-use electricity prices: New models and optimization approaches". In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 1138–1154.

[16] Antoine Legrain et al. "Combining Benders and Dantzig-Wolfe Decompositions for Online Stochastic Combinatorial Optimization". In: (2016).

[17] IBM ILOG CPLEX. "CPLEX User Manual". In: *International Business Machines Corporation* 46.53 (2009), p. 157.

[18] Scott Kirkpatrick, C Daniel Gelatt, Mario P Vecchi, et al. "Optimization by simulated annealing". In: *science* 220.4598 (1983), pp. 671–680.

[19] Junichi Mori and Vladimir Mahalec. "Planning and scheduling of steel plates production. Part II: Scheduling of continuous casting". In: *Computers & Chemical Engineering* 101 (2017), pp. 312–325.

[20] Pascal Van Hentenryck et al. "Constraint programming in OPL". In: *PPDP*. Vol. 1702. Springer. 1999, pp. 98–116.

[21] Erwin Pesch, W Domschek, and A Drexl. *Learning in Automated manufacturing: a local search approach*. Physica-Verlag, 1994.

[22] Yuji Shinano et al. "Solving hard MIPLIB2003 problems with ParaSCIP on supercomputers: An update". In: *Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International*. IEEE. 2014, pp. 1552–1561.

[23] Iiro Harjunkoski et al. "A multi-week scheduling approach for steel-making process". In: *Proc. FOCAPO*. 2003.

[24] Lixin Tang, Yang Yang, and Jiyin Liu. "An efficient optimal solution to the coil sequencing problem in electro-galvanizing line". In: *Computers & Operations Research* 37.10 (2010), pp. 1780–1796.

[25] Silvino Fernandez et al. "Scheduling a galvanizing line by ant colony optimization". In: *International Conference on Swarm Intelligence*. Springer. 2014, pp. 146–157.

[26] Carsten F Dormann and Rouven Strauss. "A method for detecting modules in quantitative bipartite networks". In: *Methods in Ecology and Evolution* 5.1 (2014), pp. 90–98.

[27] Iiro Harjunkoski and Ignacio E Grossmann. "Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods". In: *Computers & Chemical Engineering* 26.11 (2002), pp. 1533–1552.

[28] David Arthur and Sergei Vassilvitskii. "k-means++: The advantages of careful seeding". In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2007, pp. 1027–1035.

[29] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[30] Teuvo Kohonen. "Self-organized formation of topologically correct feature maps". In: *Biological cybernetics* 43.1 (1982), pp. 59–69.

[31] Hujun Yin. "The self-organizing maps: background, theories, extensions and applications". In: *Computational intelligence: A compendium*. Springer, 2008, pp. 715–762.

[32] Alan Mathison Turing. "On computable numbers, with an application to the Entscheidungsproblem". In: *Proceedings of the London mathematical society* 2.1 (1937), pp. 230–265.

[33] David E Goldberg and Jon Richardson. "Genetic algorithms with sharing for multimodal function optimization". In: *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*. Hillsdale, NJ: Lawrence Erlbaum. 1987, pp. 41–49.

[34] Gilbert Laporte. "The traveling salesman problem: An overview of exact and approximate algorithms". In: *European Journal of Operational Research* 59.2 (1992), pp. 231–247.

[35] Josef Kallrath and Anna Schreieck. "Discrete optimisation and real world problems". In: *International Conference on High-Performance Computing and Networking*. Springer. 1995, pp. 351–359.

[36] S Meeran and MS Morshed. "A hybrid genetic tabu search algorithm for solving job shop scheduling problems: a case study". In: *Journal of intelligent manufacturing* 23.4 (2012), pp. 1063–1078.

[37] Silvino Fernández et al. "Performance comparison of ant colony algorithms for the scheduling of steel production lines". In: *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM. 2015, pp. 1387–1388.

[38] Patrick David Surry. "A Prescriptive Formalism for Constructing Domain-specific Evolutionary Algorithms". PhD thesis. Citeseer, 1999.

[39] Karla L Hoffman, Manfred Padberg, and Giovanni Rinaldi. "Traveling salesman problem". In: *Encyclopedia of operations research and management science*. Springer, 2013, pp. 1573–1578.

[40] George Dantzig, Ray Fulkerson, and Selmer Johnson. "Solution of a large-scale traveling-salesman problem". In: *Journal of the operations research society of America* 2.4 (1954), pp. 393–410.

[41] David Meunier et al. "Hierarchical modularity in human brain functional networks". In: *Frontiers in neuroinformatics* 3 (2009).