POPMUSIC

29

Éric D. Taillard and Stefan Voß

Contents

Abstract

This chapter presents POPMUSIC, a general decomposition-based framework within the realm of matheuristics that has been successfully applied to various combinatorial optimization problems. POPMUSIC is especially useful

É. D. Taillard (✉)
HEIG-VD, University of Applied Sciences and Arts of Western Switzerland, Yverdon-les-Bains, Switzerland
e-mail: eric.taillard@heig-vd.ch

S. Voß
Faculty of Business Administration, Institute of Information Systems, University of Hamburg, Hamburg, Germany
e-mail: stefan.voss@uni-hamburg.de

R. Martí et al. (eds.), *Handbook of Heuristics*,
https://doi.org/10.1007/978-3-032-00385-0_31

for designing heuristic methods for large combinatorial problems that can be partially optimized. The basic idea is to optimize subparts of solutions until a local optimum is reached. Implementations of the technique to various problems show its broad applicability and efficiency for tackling especially large-size instances.

Keywords

Decomposition method · Matheuristics · Large neighborhood search · Large scale optimization

Introduction

A natural way to solve large optimization problems is to decompose them into independent subproblems that are solved with an appropriate method. However, such approaches may lead to solutions of moderate quality since the subproblems might have been created in a somewhat arbitrary fashion. While it is not necessarily easy to find an appropriate way to decompose a problem a priori, the basic idea of POPMUSIC is to locally optimize subparts of a solution, a posteriori, once a solution to the problem is available. These local optimizations are repeated until a local optimum is found (and possibly beyond). POPMUSIC stands for "Partial OPtimization Metaheuristic Under Special Intensification Conditions" [33]. It is a template, or framework, for creating heuristics repeating partial optimization on a solution. To apply this template, special conditions must be fulfilled. Among these conditions, it is supposed that a solution is composed of parts that can be optimized almost independently. Thus, POPMUSIC may be seen as a "general purpose" framework which does not need too many problem-specific adaptations if a reasonable way of decomposing a problem, or a given incumbent solution, is at hand so that it can be readily applied to a wide spectrum of problem classes.

POPMUSIC typically applies to large-size problem instances, but the template was also applied successfully to smaller instances. Typical real-life problems usually contain a very large number of elements. The POPMUSIC template proposes a solution for dealing with problem instances of that size.

Next we describe the POPMUSIC template followed by a survey of several successful applications as well as an overview on a few related approaches.

POPMUSIC Template

The idea of decomposing problems of large size into smaller subproblems easier to solve is certainly very old [8]. In the context of metaheuristics and matheuristics, this can be accomplished, for instance, by using given incumbent solutions and optimizing parts of them. In general terms, one may distinguish soft fixing and hard fixing to attain problems of reduced size that may be solved separately. While in *hard fixing* some of the decision variables of a given problem are directly excluded (e.g., by fixing them to a certain value), *soft fixing* refers to adding, say linear, constraints to the model to cut out all solutions from a problem that are beyond a certain distance from a given solution. In that way, the search can be intensified around those parts of a solution space that are not cut out.

To exemplify the POPMUSIC template in detail, we resort to the capacitated vehicle-routing problem (CVRP) as one of the first specific applications of the template [29]. We emphasize this as POPMUSIC can be well illustrated on the CVRP. In its simplest version, it consists of finding a set of vehicle tours, starting from a depot, servicing customers asking given quantities of goods and coming back to the depot. Knowing the vehicle capacity and all distances between clients, a set of tours, visiting each customer exactly once, is searched in such a way that the total distance of the tours is minimized.

If a solution is available, a tour can be considered as a part of the solution. Each tour can be interpreted as being independent from the others and can be optimized independently. Optimizing a tour consists of finding an optimal trip (or tour) of a traveling salesman problem (TSP). Since the number of customers on a tour of a CVRP is limited and exact codes for the traveling salesman problem are available and efficient up to few dozens of cities and beyond [9], the optimization of single CVRP tours can be considered as a simple task.

Now, if we consider the subset of customers delivered by a subset of tours of a given solution, it is also possible to optimize independently the delivery of this subset of customers. So, a subset of parts of a solution creates a subproblem that can be optimized independently from the remaining of the problem. For a CVRP, such a subproblem is a CVRP of reduced size.

If an optimization method is available that can treat CVRPs with up to r tours efficiently, larger instances can be tackled by repeating optimizations on subproblems containing up to r tours. The POPMUSIC template suggests a tactic for building subproblems that can be potentially improved. Instead of generating any subproblem with r parts, as done by matheuristics like large neighborhood search (LNS), POPMUSIC introduces a notion of proximity between parts. Subproblems are built by first selecting a part called a *seed part* and r of its closest parts. Figure 1 illustrates the principle of the generation of a subproblem containing six tours.

Given a solution composed of p parts, there are only p subproblems composed of $r < p$ parts that can be so built, instead of the $\frac{p!}{(p-r)!\cdot r!}$ possible ones. In contrast to the LNS template where the user must choose at each step how to build a subproblem and when to stop the optimization process, POPMUSIC-based approaches have a natural stopping criterion: When the p subproblems are solved optimally or with satisfaction, the process can be stopped.

To be specific, in the POPMUSIC template, it is supposed that a solution s can be decomposed into p parts $s_1, \dots, s_p$ and that a proximity measure between two parts has been defined. Let U with $|U| =: p$ be the set of parts that have not been used as seed part for building a subproblem. Initially, U contains all p parts and U

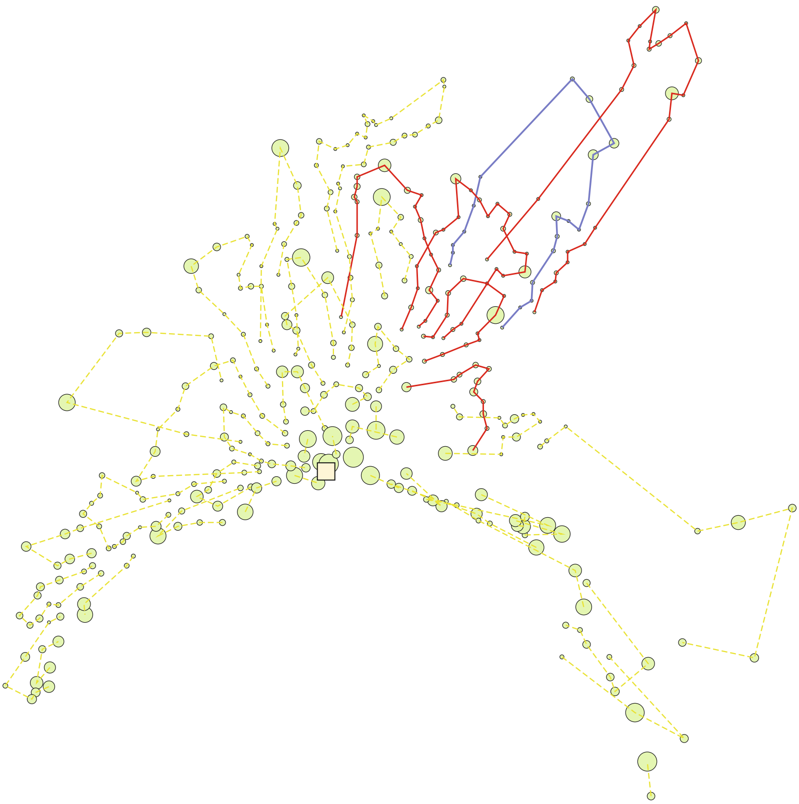

Fig. 1 For the vehicle-routing problem, the definition of a part can be a vehicle tour. In this figure, the size of the disks represents the customers' demand, and the trips from and to the depot (square) are not drawn. Here, the proximity between parts is measured by the distance of their center of gravity. A subproblem is a smaller VRP composed of the customers serviced by six tours

is empty at the end. Let $m(R)$ be an optimization method that allows to optimize a subproblem R composed of r parts. With these hypothesis, the POPMUSIC template can be expressed by Algorithm 1.

POPMUSIC can be seen from different perspectives: As mentioned above, it can be considered as a large neighborhood search [26] or as a soft fixing approach. In this case, the neighborhood considered consists of finding the best way to reshape the elements of r parts. The neighborhood is qualified as large, in contrast to local search that modifies only few elements of a solution at each step, allowing explicitly a complete examination of the neighborhood. When all the elements of r parts must be optimized, it is not possible to proceed to an explicit examination of the whole neighborhood, since its size is much too large. So, either an exact method is used, making POPMUSIC a matheuristic (see [22]), or an approximate algorithm is used, typically based on a metaheuristic.

Algorithm 1: POPMUSIC template

Input: Initial solution s composed of p parts $s_1, \dots, s_p$, sub-problem optimization method m
Output: Improved solution s
$U = \{s_1, \dots, s_p\}$
while $U \neq \emptyset$ **do**
    Select $s_g \in U$ // s_g: seed-part
    Build sub-problem R with the r parts of s that are the closest to s_g
    Optimize R with method m
    if R *improved* **then**
        Update solution s
        Remove from U the parts that do not belong to s anymore
        Insert in U the parts of optimized sub-problem R
    else // R not improved
        Remove s_g of U
    end
end

Deriving a specific POPMUSIC implementation is described by means of a specific application in the next section.

POPMUSIC Adaptation for Location Routing

The POPMUSIC template requires the programmer to define five components: the *initial solution*, a *part*, the *proximity* between parts, the *seed-part* choice, and an *optimization method*. This section illustrates how these components can be implemented for a location-routing problem (LRP). Very briefly, an LRP is a vehicle-routing problem with several depots where the location of the depots must be decided in addition to the building of vehicle tours.

Initial Solution

A key point for the treatment of large problem instances is to analyze the algorithmic complexity of each step of the solution method. Examining the POPMUSIC template, the first question is about the number of times the `while` loop at line 1 is repeated. Numerical experiments on different problems—unsupervised clustering [30], map labeling [3], location routing [4], berth allocation [17, 20]—have shown that this loop is repeated a number of times that grows quasi-linearly with the problem size. This behavior is typical for local searches based on standard neighborhoods where the theoretical exponential complexity is not observed. To lower the algorithmic complexity, the other steps of the template must be carefully designed.

The step that looks most complex is naturally the optimization of subproblems at line 1. Indeed, the computational time for optimizing a subproblem with r parts grows very fast with r, for instance, exponentially if an exact method is used.

The user can control the computational time by modifying the unique parameter r of POPMUSIC template. Let us suppose that the user has fixed the value of r to a value for which the subproblems are sufficiently large for getting solutions of good quality while limiting the computational time of the optimization method. If r is fixed, solving a subproblem takes a constant time—maybe large, but constant. So, if the use of an efficient optimization method is a key point of the POPMUSIC template, this is not an issue for limiting the algorithmic complexity.

The choice of the seed part can be implemented in constant time, for instance, by making a random choice or by storing set U as a stack or a queue. So, the key points for limiting the algorithmic complexity of a POPMUSIC implementation is to use an appropriate technique for producing the initial solution and for building subproblems from a seed part. Indeed, without an appropriate data structure, building a subproblem can take a time proportional to the problem size, leading to a global complexity being at least quadratic.

Alvim and Taillard [4] have presented how to build in $O(n^{3/2})$ a solution to a location-routing problem while building a data structure allowing to build a subproblem in $O(r)$. The idea of the construction is the following: Let us suppose that it is possible to advise a distance measure between the entities of the problem. For the LRP, this distance is part of the problem data: It is simply the distance between customers. The entities of the problem are grouped into $\sqrt{n}$ clusters by means of a heuristic method for the p-median problem. Although NP-hard, the p-median problem can be approximately solved with an empirical complexity of $\overline{O}((p \cdot n + (\frac{n}{p})^2)$, where p is the number of clusters desired. In this problem, the clusters are built by choosing p central elements and assigning all remaining elements to the closest center. By choosing $p = \sqrt{n}$, it is possible to cut a problem of size n into $\sqrt{n}$ clusters containing approximately $\sqrt{n}$ elements in $\overline{O}((n^{3/2})$. This method is illustrated in Fig. 2.

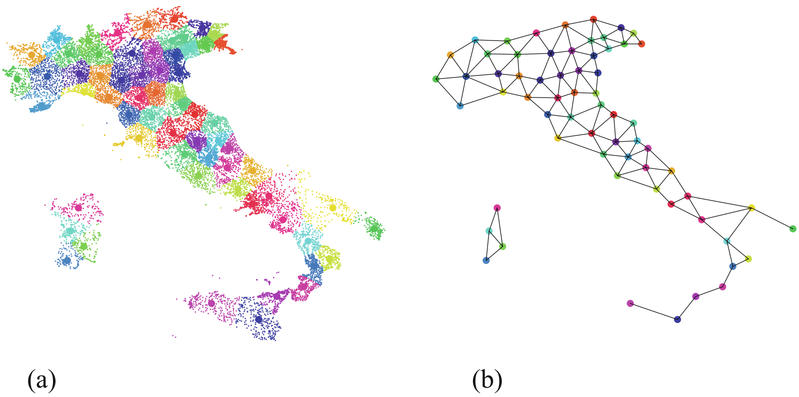

Fig. 2 Illustration of the decomposition of a problem into $\sqrt{n}$ clusters and building a proximity network between the clusters. (**a**) Problem decomposition into clusters. (**b**) Building proximity network

Then, all centers can be compared 2 by 2 to create proximity relations. Figure 2b illustrates the proximity relation obtained by considering that clusters a and b are neighbors if an entity of one of these clusters has the center of the other one as second closest center. Each cluster can then be decomposed into $O(\sqrt{n})$ clusters to create a total of $O(n)$ small clusters containing a constant number of elements. So, it is possible to create proximity relations between every entity of a problem of size n in $O(n^{3/2})$. In the context of the LRP, a vehicle tour can be assigned to the entities of a small cluster. The proximity relations can be exploited in POPMUSIC for finding efficiently the r parts that are the closest from a seed part.

However, if no versatile method for finding incumbent solutions is at hand, any type of heuristic or even a randomized approach may be used to derive them.

Part and Subproblem Definitions

The definition of a part strongly depends on the problem under consideration and on the procedure available for optimizing subproblems. For vehicle-routing problems, [4, 24, 29] have defined the customers serviced by a vehicle tour as a part. This definition is justified by the fact that the subproblems built around a seed part are smaller VRPs or multi-depot VRPs that can be efficiently optimized with a metaheuristic-based algorithm, like a basic tabu search or genetic algorithm hybridized with a variable neighborhood search. Several definitions have been adopted for the proximity between parts: For Euclidean instances, the distance between the center of gravity of tours can be used; generally, the technique presented in section "Initial Solution" can be used for defining the proximity.

Seed-Part Selection

Very few works have studied the impact of the seed-part selection procedure. In many applications, subproblems are generated around a seed part arbitrarily chosen in U (next available, random one, ...). Preliminary results in [4] have shown that a set U managed as a stack (last part entered in U is selected) seems to be better than an arbitrary selection.

Empirical Complexity

In [4], problem instances 10000 times larger than those usually treated in the literature have been tackled. As mentioned above, the most complex part is the generation of an initial solution in $O(n^{3/2})$. The subproblem optimization is less complex, almost linear. Figure 3 shows the evolution of the computational effort as a function of problem size for different POPMUSIC phases. The increase of computational time is higher for the steps of generating the initial solution (solving a p-median problem with two levels, as presented in section "Initial Solution") than

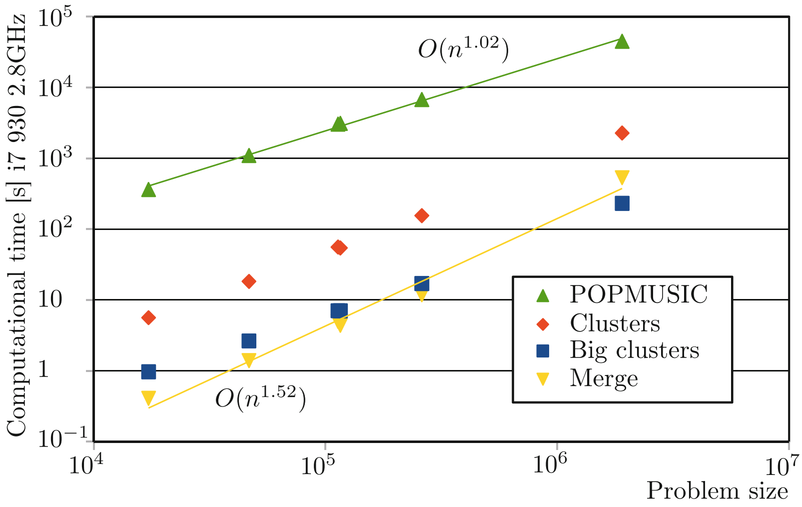

Fig. 3 Evolution of the computational effort as a function of problem size for various parts of the algorithm: building large clusters, decomposing large clusters into small ones containing subsets of customers that can be delivered by the same vehicle, finding positions for the depots and optimizing subproblems

for the subproblem optimization. However, this last phase still takes most of the computational effort, even with instances with two millions of entities.

POPMUSIC Applications

While we have used the CVRP and a location-routing problem to clarify concepts, in this section, we summarize a few additional successful applications of POPMUSIC. Other applications not specifically mentioned refer, e.g., to the turbine runner balancing problem [33] or the p-cable trench problem [19].

Application of POPMUSIC as Matheuristic for the CVRP

Recently, [27] precisely adapted POPMUSIC for the capacitated vehicle-routing problem (CVRP) using a branch-cut-and-price algorithm, which combines column and cut generation with several additional mechanisms to optimize the subproblems. These authors propose a variant for constructing subproblems:

Each customer takes turns playing the role of a seed. Then, all other customers are listed in ascending order based on their distance from the seed customer. As long as the subproblem contains a number of customers not exceeding a threshold α, the route containing the next closest customer to the seed is added to the subproblem—provided it is not already included. Thus, the parameter r, which fixes the number of routes, is replaced by α, which sets the maximum number of customers in a subproblem.

To avoid unnecessary optimization attempts, the set of previously optimized subproblems is stored. Additionally, a variant of the POPMUSIC framework is proposed in this chapter, with the idea of increasing α by a quantity δ once each customer has played the role of a seed.

POPMUSIC has also been adapted to a variant of the CVRP, namely the multi-depot cumulative capacitated vehicle-routing problem, by Lalla-Ruiz and Voß [18].

Application of POPMUSIC to the TSP

A very simple way to apply the POPMUSIC framework to the traveling salesman problem (TSP) is to consider each city as a part and measure proximity between two cities by the number of edges that separate them in the tour to be optimized.

With this definition, a subproblem is a path that can be optimized while keeping its two endpoint cities fixed. This approach was studied in [32].

This proximity definition allows for the re-optimization of a tour with a linear empirical computational effort. However, its drawback is that it cannot "untangle" crossings between edges separated by a number of cities greater than the subproblem size.

The biggest challenge in applying POPMUSIC to the TSP is, therefore, obtaining an initial solution that lends itself well to further optimization. Taillard [31] proposed an $O(n \log n)$ approach that generates solutions for gigantic instances (with billions of cities). While the solution quality is relatively modest, it is significantly better than that of the nearest-neighbor heuristic, for example.

The main advantage of this method is its ability to quickly identify a very limited subset of edges that include those of excellent solutions, even for non-Euclidean instances. This property is leveraged in the latest versions of the LKH software [14, 15], which are currently among the best freely available solvers for this class of problems.

Application of POPMUSIC to Clustering

The POPMUSIC template is particularly well adapted to solve large clustering problem instances. The goal of clustering is to create groups of entities as well separated as possible while containing entities as homogeneous as possible. This means that metrics are available for measuring the similarity or the dissimilarity between elements. If the number of clusters is relatively large, using the POPMUSIC template is interesting. In this case, a part of a solution can be constituted as the set of entities belonging to the same cluster. The measure of the separation between groups can be used for defining the proximity between parts.

This technique was successfully applied to unsupervised clustering with centroids (sum-of-squares clustering, p-median, multisource Weber) in [30]. Figure 4 illustrates the construction of a subproblem from a solution to a p-median problem.

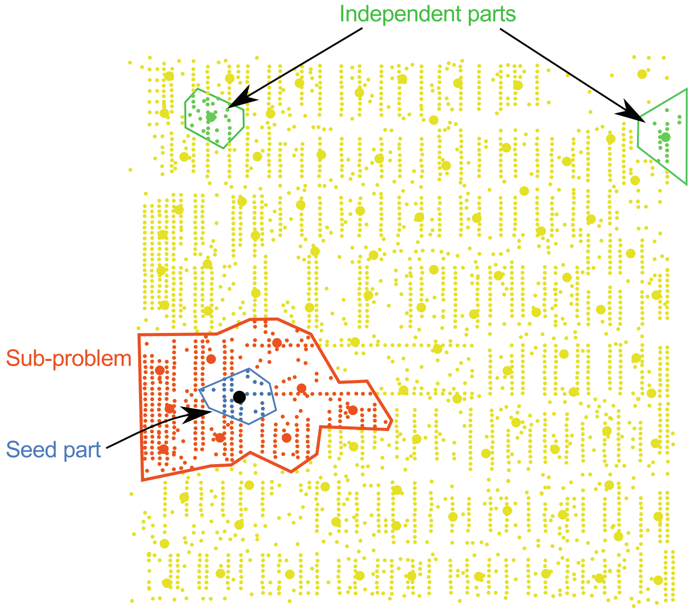

Fig. 4 Building a subproblem from a seed part in the case of a p-median problem. Clusters that are well separated can be considered as independent and should not be involved in the same subproblem

Application of POPMUSIC to Max Weight Stable Set Problems

The POPMUSIC template is also well adapted to the search of a stable set with maximum weight in a graph. Several practical problems can be modeled under this form. For instance, the map labeling problem can be formulated as a stable set with maximum weight. The problem is the following: Given objects on the Euclidean plane, one wants to place a label around them to identify them. For instance, when drawing a map, an object can be the top of a mountain and the label the name of the mountain. The label can be placed at several predefined positions. The problem consists of finding a position for each label in such a way that no label partially covers another one. If it is not possible to label all objects without superposition, it is searched to maximize the number of labels correctly placed. For typographic reasons, a different weight is associated to each label position, reflecting the positioning preferences or the importance of the object.

The problem can be formulated as the search of a stable set with maximum weight in a graph. The graph is built as follows: A node is associated with every label position, with a weight corresponding to the preference. An edge connects two nodes if the corresponding labels are incompatible: Either they are associated with the same object (every object must be labeled at most once) or they overlap. Figure 5 illustrates the construction of a graph for labeling three objects with four possible label positions for each of them.

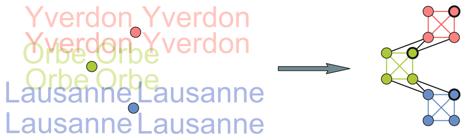

Fig. 5 Formulation of the problem of map labeling under a stable set

Applying the POPMUSIC template to this problem can be done as follows: A part is an object to label, so the set of all nodes representing the different possible label positions for this object. The distance between two objects (interpreted as parts) is the minimum number of edges needed to connect two nodes associated to labels of the objects they identify. A subproblem is constituted by r objects for which the label position can be modified. A subproblem must also consider other labels of the current solution that may overlap, but without searching to modify the position of these labels.

The optimization procedure is a re-implementation of a tabu search due to [35] for which the parameters were tuned for solving at best problem instances with few dozens of objects. Alvim and Taillard [2] have shown that excellent solutions can be obtained for problem instances with one thousand objects and that the method can still be used for instances with millions of objects.

This work was continued by Laurent et al. [21] for allowing the labeling of cartographic maps containing non-punctual objects like lines (streets, rivers) or polygons (states, countries). The subproblem optimization used in this reference is based on ejection chains. The algorithms developed by Laurent et al. [21] have been integrated in the QGIS open-source software (http://www.qgis.org/en/site/).

Various problems in the domain of transportation can be formulated as a map labeling (or a maximum weight stable set). The assignment of cruising levels of commercial aircraft is one of them. Knowing the departure hour and horizontal trajectory of every aircraft at a continent level, it is required to assign each of them a cruising level in order to avoid the collision between aircraft. The label shape is determined in this case by the possible position of the aircraft for a given time interval. The exact position of the aircraft cannot be determined, since the departure hour may fluctuate (traffic delay in the airport) and the weather conditions may affect the speed and position of the aircraft (wind, clouds, storm, ...). For a time interval, it is required to find a level for each label in such a way that no label covers another one.

Application of POPMUSIC to Berth Allocation Problems

The berth allocation problem (BAP) is a well-known optimization problem within maritime shipping. It aims at assigning and scheduling incoming vessels to berthing positions along the quay of a container terminal. Lalla-Ruiz and Voß[17] propose two POPMUSIC approaches that incorporate an existing mathematical programming formulation based on modeling the problem as a set partitioning problem. The computational experiments reveal state-of-the-art results outperforming all previous approaches regarding solution quality.

To be specific, [17] study the application of POPMUSIC for solving the discrete dynamic berth allocation problem (DBAP). In the DBAP, one is given a set of incoming container vessels, N, and a set of berths, M. Each container vessel, $i \in N$, must be assigned to an empty berth, $k \in M$, within its time window, $[t_i, t_i']$, and the assigned berth time window, $[s_k, e_k]$. A simplified assumption is that each berth can handle at most one vessel at a time. For each container vessel, $i \in N$, its handling time, ρ_{ik}, depends on the berth $k \in M$ where it is assigned to. That is, the service time of a given vessel differs from one berth to another. Moreover, some vessels may have forbidden berths in order to model water-depth or maintenance constraints. Finally, each vessel $i \in N$ has a given service priority, denoted as v_i, according to its contractual agreement with the terminal.

A natural way to define parts is by means of the berths themselves. If a restricted number of berths (and assigned vessels) can be handled with an exact approach, this allows for efficient subproblem solving. Moreover, it is also easy to define the proximity between berths, taking real distances.

An interesting observation deduced from solving the DBAP and extensions by means of POPMUSIC also reveals some general lessons learned regarding the definition of parts. Let us assume a problem extension as the berth allocation problem under time-dependent limitations (BAP-TL) [20]. In this problem, in order to assign a vessel to a berth, terminal managers might have to take into account not only the berthing place and vessel draft, but also the arrival and berthing time of an incoming vessel while observing changing environments due to tidal changes. Note that in the DBAP the decisions about to which berth the vessel should be assigned to is relevant since there are different handling times depending on the berth. This is relaxed in the BAP-TL, i.e., all the berths provide the same handling time, having also an important implication on the proper definition of parts. Note also that in the BAP-TL the berthing time is important due to the tidal constraints. Thus, it makes sense to define the parts not necessarily following the berths delimitation, but defining the parts as intervals of time. Numerical results are given in [20].

To end this subsection, we should point to an interesting analogy which may be drawn between the pure (and possibly simplified) berth allocation problem and the map labeling problem. In this case, the width of a label corresponds to the time interval during which the vessel must be at the berth for being loaded/unloaded. The height of the label is the vessel length. A label can be placed at different positions along the berth. Figure 6 provides an example of the transformation of a small problem with four vessels arriving at different times and that can be placed at few different positions along a berth (considering vessel length and time windows, TW).

The POPMUSIC template was used in [17, 20] for the dynamic berth allocation problem without and with the consideration of tidal constraints. In these references, the optimization procedure is an exact method.

The use of the POPMUSIC framework in multi-objective optimization for an integrated berth allocation and quay crane assignment problem is described in [36].

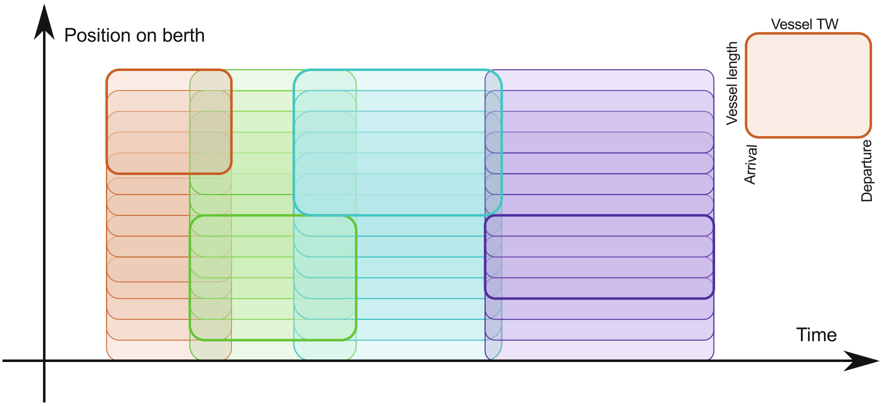

Fig. 6 Transformation of berth allocation problem to a map labeling

An integrative algorithm framework is developed, composed of POPMUSIC, the Strength Pareto Evolutionary Algorithm 2, and k-means clustering. An extension, again applying the POPMUSIC framework in multi-objective optimization, toward the adoption of different green technologies is described in [37].

Related Approaches

For large optimization problems, it is often possible to interpret a solution as composed of parts or chunks [34] as can also be found under the term vocabulary building. Suppose that a solution can be represented as a set of parts as seen above, some parts are more in relation with some other parts so that a corresponding heuristic measure can be defined between two parts.

The *corridor method* (CM) has been presented by Sniedovich and Voß [28] as a hybrid metaheuristic, linking mathematical programming techniques with heuristic schemes. The basic idea of the CM relies on the use of an exact method over restricted portions of the solution space of a given problem. Given an optimization problem, the basic ingredients of the method are a very large feasible space, and an exact method that could easily solve the problem if the feasible space were not large.

The basic concept of a *corridor* is introduced to delimit a portion of the solution space around the incumbent solution. The optimization method will then be applied within the neighborhood defined by the corridor with the aim of finding an improved solution. Consequently, the CM defines method-based neighborhoods, in which a neighborhood is built taking into account the method used to explore it.

As mentioned before, POPMUSIC may be seen as a soft fixing approach as does the CM. Food for thought might be considered putting POPMUSIC into perspective regarding other methods like large neighborhood search, adaptive randomized decomposition, kernel search, etc. An older research which may serve as motivation relates to solving the job shop problem by means of the shifting bottleneck heuristic [1]. The idea of developing heuristics identifying a small/moderate size subset of variables in order to intensify the search in a promising region of the solution space has been used in other contexts. In the knapsack problem family, for instance, [7] propose the idea of selecting a small subset of items (called the core) and solving exactly a restricted problem on that subset. The use of an expanding method to modify the size of the core during the algorithm execution is proposed by Pisinger [25]. A heuristic framework called kernel search has been proposed for the solution of mixed-integer linear problems in [5, 6]. The kernel search framework is based on the idea of exhaustively exploring promising portions of the solution space. Kernel search is similar to POPMUSIC since "buckets," analogous to parts, are defined at the beginning of the search. During the execution, the algorithm revises the definition of the core problem, called "kernel," by adding/removing buckets to/from the current problem. Once the current kernel is defined, an exact method is applied to the restricted problem.

Local branching [10] is another soft fixing technique, in which the introduction of linear constraints is used to cut solutions from the feasible space. The feasible space of the constrained problem includes only solutions that are not too far away from the incumbent. When presented, local branching was used within a branch and bound framework. A generalized local branching concept has been proposed by Hill and Voß [16].

Variable neighborhood decomposition search is in line with the strategy employed by POPMUSIC. Neighborhoods around an incumbent are defined using a distance metric and explored via any method. If a local optimum better than the incumbent is found in the current neighborhood, the neighborhood is re-centered around the new incumbent, and the mechanism moves on to the exploration of the new neighborhood. If no improvement occurs, the search moves to the next neighborhood defined around the same incumbent. An interesting modification of variable neighborhood decomposition is provided in [13], where variable neighborhood search is coupled with local branching. Neighborhoods are defined imposing linear constraints on a mixed-integer linear programming model, as done in local branching, and then explored using a general-purpose solver as black box.

Fischetti et al. [11] propose a similar method called diversification, refining, and tight refining (DRT). It is aimed at solving problems with two-level variables, in which fixing the value of the first-level variables leads to an easier to solve, but still hard, subproblem. Finally, another way to see POPMUSIC is to exploit the proximate optimality principle (POP, see, e.g., [12]). In tabu search, the POP notion is exploited by performing a number of iterations at a given level before restoring the best solution found to initiate the search at the next level. In that context, a level corresponds to the optimization of a subproblem in the POPMUSIC terminology.

POPMUSIC is also applied toward various combinatorial optimization problems as part of an integrated approach combining various metaheuristics. As an example, we mention [23] for a matheuristic approach hybridizing particle swarm optimization, simulated annealing, and mathematical programming based on the POPMUSIC concept.

Conclusion and Future Works

The main strength of POPMUSIC is its simplicity, the fact that it has a unique parameter and its ability to solve large problem instances. So, the main effort when implementing a POPMUSIC-based procedure is not devoted to parameter tuning as it can be the case for other metaheuristics. But there is no free lunch. The weakness of POPMUSIC is that an initial solution with a structure adapted to the template must be available as well as an optimization procedure for the subproblems. These points make POPMUSIC less general and more problem-specific than other metaheuristics. Adapting POPMUSIC to various hard problems which are not easy to decompose is a first research avenue.

Up to now, very few works have been devoted to study the influence of POPMUSIC options, such as the way the seed part is chosen or the procedure used for optimizing subproblems. These are other promising research topics for the future. It can also be interesting to revisit existing methods that are somewhat based on decomposition principles under the form of the POPMUSIC template. This could lead to simplifications and/or efficiency improvements.

The POPMUSIC template can be adapted to parallel implementations. Our first works on the vehicle-routing problem were primarily devoted to parallel implementations of metaheuristics. A relatively easy issue is to use the few cores of CPU in parallel. However, implementing a POPMUSIC with a number of processors dependent on problem size is less trivial. Another research avenue is to see how to exploit the large number of processors of graphic process units in the context of POPMUSIC.

Cross-References

▶ Iterated Local Search
▶ Iterated Greedy
▶ Tabu Search

Competing Interest Declaration The author(s) has no competing interests to declare that are relevant to the content of this manuscript.

References

1. Adams J, Balas E, Zawack D (1988) The shifting bottleneck procedure for job shop scheduling. Manag Sci 34:391–401
2. Alvim ACF, Taillard ÉD (2007) An efficient POPMUSIC based approach to the point feature label placement problem. In: Metaheuristic international conference (MIC'07) proceedings
3. Alvim ACF, Taillard ÉD (2009) POPMUSIC for the point feature label placement problem. Eur J Oper Res 192(2):396–413. https://doi.org/10.1016/j.ejor.2007.10.002
4. Alvim ACF, Taillard ÉD (2013) POPMUSIC for the world location routing problem. EURO J Transport Logist 2:231–254

5. Angelelli E, Mansini R, Speranza MG (2010) Kernel search: a general heuristic for the multi-dimensional knapsack problem. Comput Oper Res 37(11):2017–2026
6. Angelelli E, Mansini R, Speranza MG (2012) Kernel search: a new heuristic framework for portfolio selection. Comput Optim Appl 51(1):345–361
7. Balas E, Zemel E (1980) An algorithm for large zero-one knapsack problems. Oper Res 28(5):1130–1154
8. Ball MO (2011) Heuristics based on mathematical programming. Surveys Oper Res Manag Sci 16(1):21–38. https://doi.org/10.1016/j.sorms.2010.07.001
9. Concorde (2015) Concorde TSP solver. http://www.math.uwaterloo.ca/tsp/concorde/index.html
10. Fischetti M, Lodi A (2003) Local branching. Math Programm B 98:23–47
11. Fischetti M, Polo C, Scantamburlo M (2004) A local branching heuristic for mixed-integer programs with 2-level variables, with an application to a telecommunication network design problem. Networks 44(2):61–72
12. Glover F, Laguna M (1997) Tabu search. Kluwer, Dordrecht
13. Hansen P, Mladenović N, Urosević D (2006) Variable neighborhood search and local branching. Comput Oper Res 33(10):3034–3045
14. Helsgaun K (2022). Helsgaun's implementation of Lin-Kernighan. http://webhotel4.ruc.dk/~keld/research/LKH/. Version LKH-2.0.10
15. Helsgaun K (2024) Extension of Helsgaun's implementation of Lin-Kernighan for constrained TSP and VRPs'. http://webhotel4.ruc.dk/~keld/research/LKH-3/. Version LKH-3.0.10
16. Hill A, Voß S (2018) Generalized local branching heuristics and the capacitated ring tree problem. Discrete Appl Math 242:34–52. https://doi.org/10.1016/j.dam.2017.09.010
17. Lalla-Ruiz E, Voß S (2016) POPMUSIC as a matheuristic for the berth allocation problem. Ann Math Artif Intell 76:173–189
18. Lalla-Ruiz E, Voß S (2020) A POPMUSIC approach for the multi-depot cumulative capacitated vehicle routing problem. Optim Lett 14(3):671–691. https://doi.org/10.1007/s11590-018-1376-1
19. Lalla-Ruiz E, Schwarze S, Voß S (2016) A matheuristic approach for the p-cable trench problem. Lect Notes Comput Sci 10079:247–252
20. Lalla-Ruiz E, Voß S, Exposito-Izquierdo C, Melian-Batista B, Moreno-Vega JM (2017) A POPMUSIC-based approach for the berth allocation problem under time-dependent limitations. Ann Oper Res 253:871–897
21. Laurent M, Taillard ÉD, Ertz O, Grin F, Rappo D, Roh S (2009) From point feature label placement to map labelling. In: Metaheuristic international conference (MIC'09) Proceedings
22. Maniezzo V, Stützle T, Voß S (eds) (2009) Matheuristics: hybridizing metaheuristics and mathematical programming. Springer, Berlin
23. Nourmohammadzadeh A, Voß S (2023) An effective matheuristic approach for robust bus driver rostering with uncertain daily working hours. Lect Notes Comput Sci 14239:365–380
24. Ostertag A, Doerner KF, Hartl RF, Taillard ÉD, Waelti P (2009) POPMUSIC for a real-world large-scale vehicle routing problem with time windows. J Oper Res Soc 60(7):934–943. https://doi.org/10.1057/palgrave.jors.2602633
25. Pisinger D (1999) Core problems in knapsack algorithms. Oper Res 47(4):570–575
26. Pisinger D, Ropke S (2007) A general heuristic for vehicle routing problems. Comput Oper Res 34(8):2403–2435. https://doi.org/10.1016/j.cor.2005.09.012
27. Queiroga E, Sadykov R, Uchoa E (2021) A POPMUSIC matheuristic for the capacitated vehicle routing problem. Comput Oper Res 136:105475. https://doi.org/10.1016/j.cor.2021.105475
28. Sniedovich M, Voß S (2006) The corridor method: a dynamic programming inspired metaheuristic. Control Cyber 35:551–578
29. Taillard ÉD (1993) Parallel iterative search methods for vehicle routing problems. Networks 23(8):661–673. https://doi.org/10.1002/net.3230230804
30. Taillard ÉD (2003) Heuristic methods for large centroid clustering problems. J Heurist 9(1):51–73

31. Taillard ÉD (2022) A linearithmic heuristic for the travelling salesman problem. Eur J Oper Res 297(2):442–450. https://doi.org/10.1016/j.ejor.2021.05.034
32. Taillard ÉD, Helsgaun K (2019) POPMUSIC for the travelling salesman problem. EURO J Oper Res 272(2):420–429. https://doi.org/10.1016/j.ejor.2018.06.039
33. Taillard E, Voß S (2002) POPMUSIC — partial optimization metaheuristic under special intensification conditions. In: Ribeiro CC, Hansen P (eds) Essays and surveys in metaheuristics. Kluwer, Boston, pp 613–629. https://doi.org/10.1007/978-1-4615-1507-4_27
34. Woodruff DL (1998) Proposals for chunking and tabu search. Eur J Oper Res 106:585–598
35. Yamamoto M, Camara G, Lorena LAN (2002) Tabu search heuristic for point-feature cartographic label placement. GeoInformatica 6:77–90
36. Yu J, Voß S, Song X (2022) Multi-objective optimization of daily use of shore side electricity integrated with quayside operation. J Cleaner Prod 351:131406. https://doi.org/10.1016/j.jclepro.2022.131406
37. Yu J, Tang G, Voß S, Song X (2023) Berth allocation and quay crane assignment considering the adoption of different green technologies. Transport Res Part E Logist Transport Rev 176:103185. https://doi.org/10.1016/j.tre.2023.103185