LNCS 4771

Thomas Bartz-Beielstein
María José Blesa Aguilera
Christian Blum
Boris Naujoks
Andrea Roli
Günter Rudolph
Michael Sampels (Eds.)

Hybrid Metaheuristics

4th International Workshop, HM 2007
Dortmund, Germany, October 2007
Proceedings

Springer

Thomas Bartz-Beielstein
María José Blesa Aguilera
Christian Blum   Boris Naujoks
Andrea Roli   Günter Rudolph
Michael Sampels (Eds.)

Hybrid Metaheuristics

4th International Workshop, HM 2007
Dortmund, Germany, October 8-9, 2007
Proceedings

Springer

Volume Editors

Thomas Bartz-Beielstein
Fachhochschule Köln, Germany, E-mail: Thomas.Bartz-Beielstein@fh-koeln.de

María José Blesa Aguilera
Universitat Politècnica de Catalunya, Spain, E-mail: mjblesa@lsi.upc.edu

Christian Blum
Universitat Politècnica de Catalunya, Spain, E-mail: cblum@lsi.upc.edu

Boris Naujoks
University of Dortmund, Germany, E-mail: boris.naujoks@uni-dortmund.de

Andrea Roli
University of Bologna, Italy, E-mail: andrea.roli@unibo.it

Günter Rudolph
Universität Dortmund, Germany, E-mail: guenter.rudolph@uni-dortmund.de

Michael Sampels
Université Libre de Bruxelles, Belgium, E-mail: msampels@ulb.ac.be

Library of Congress Control Number: 2007936297

CR Subject Classification (1998): F.2, F.1, G.1.6, G.1.2, G.2.1, I.2

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-540-75513-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-75513-5 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12170852 06/3180 5 4 3 2 1 0

Preface

The International Workshop on Hybrid Metaheuristics is now an established event and reaches its fourth edition with HM 2007. One of the main motivations for initiating it was the need for a forum to discuss specific aspects of hybridization of metaheuristics. Hybrid metaheuristics design, development and testing require a combination of skills and a sound methodology. In particular, comparisons among hybrid techniques and assessment of their performance have to be supported by a sound experimental methodology, and one of the mainstream issues of the workshop is to promote agreed standard experimental methodologies. These motivations are still among the driving forces behind the workshop and, in these four years, we have observed an increasing attention to methodological aspects, from both the empirical and theoretical sides. The papers selected for presentation at HM 2007 are indeed a representative sample of research in the field of hybrid metaheuristics. They range from methodological to application papers. Moreover, some of them put special emphasis on the experimental analysis and statistical assessment of results.

Among the publications in this selection, there are some that focus on the integration of metaheuristics with mathematical programming, constraint satisfaction or machine learning techniques. This interdisciplinary subject is now widely recognized as one of the most effective approaches for tackling hard problems, and there is still room for new results. To achieve them, the community needs to be open toward other research communities dealing with problem solving, such as those belonging to artificial intelligence (AI) and operations research (OR).

We also note that the use of software libraries for implementing metaheuristics is increasing, even though we have to observe that the users of a software library are usually its developers, thus reducing the advantages in terms of software design and development. We believe that this situation is going to change in favor of a scenario in which some libraries will be used by most metaheuristic developers.

Finally, there are also some works describing applications of metaheuristics in continuous optimization. The cross-fertilization between combinatorial and continuous optimization is extremely important, especially because many real-world problems can be naturally modeled as mixtures of discrete and continuous components.

It is already a tradition of the workshop to keep the acceptance rate of papers relatively low: this makes it possible to publish official proceedings, which can be taken as one of the main references in the field. Besides this, special care is taken with respect to the reviewing process, during which the authors are provided with constructive and detailed reviews. For this reason, the role of the Program Committee members is crucial, and we are very grateful to them for the effort they made examining the papers and providing detailed reviews. Among the 37 submissions received, 14 papers have been selected on the basis of the Program Committee members' suggestions. We are further grateful to Catherine C. McGeoch and Thomas Stützle, who both accepted our invitation to give an overview talk.

Looking back to the previous editions of the workshop, we observe a positive trend concerning experimental methodology. Moreover, some topics, such as the integration of metaheuristics with OR and AI techniques, have become established themes. We believe that a grounded discipline in hybrid metaheuristics could bring advantages in problem solving in many areas, such as constrained optimization, mixed integer optimization and also stochastic and online problems, which are probably one of the new frontiers still to be fully explored.

August 2007

Thomas Bartz-Beielstein
María J. Blesa
Christian Blum
Boris Naujoks
Andrea Roli
Günter Rudolph
Michael Sampels

Organization

General Chair

Günter Rudolph	Universität Dortmund, Germany

Program Chairs

Thomas Bartz-Beielstein	Fachhochschule Köln, Germany
María J. Blesa	Universitat Politécnica de Catalunya, Barcelona, Spain
Christian Blum	Universitat Politécnica de Catalunya, Barcelona, Spain
Boris Naujoks	Universität Dortmund, Germany
Andrea Roli	Università di Bologna, Italy
Michael Sampels	Université Libre de Bruxelles, Belgium

Local Organization

Nicola Beume	Universität Dortmund, Germany
Gundel Jankord	Universität Dortmund, Germany
Maria Kandyba	Universität Dortmund, Germany

Program Committee

Dirk Arnold	Dalhousie University, Canada
Mauro Birattari	Université Libre de Bruxelles, Belgium
Jürgen Branke	AIFB, Universität Karlsruhe, Germany
Marco Chiarandini	Syddansk Universitet, Denmark
Francisco Chicano	Universidad de Málaga, Spain
Carlos Cotta	Universidad de Málaga, Spain
Luca Di Gaspero	Università di Udine, Italy
Marco Dorigo	Université Libre de Bruxelles, Belgium
Michael Emmerich	Universiteit Leiden, The Netherlands
Thomas Jansen	Universität Dortmund, Germany
Joshua Knowles	University of Manchester, UK
Andrea Lodi	Università di Bologna, Italy
Vittorio Maniezzo	Università di Bologna, Italy
Daniel Merkle	Universität Leipzig, Germany
Bernd Meyer	Monash University, Australia
Martin Middendorf	Universität Leipzig, Germany

José A. Moreno	Universidad de La Laguna, Spain
J. Marcos Moreno	Universidad de La Laguna, Spain
David Pelta	Universidad de Granada, Spain
Steven Prestwich	4C, Cork, Ireland
Mike Preuss	Universität Dortmund, Germany
Christian Prins	Université de Technologie de Troyes, France
Günther Raidl	Technische Universität Wien, Austria
Andreas Reinholz	Universität Dortmund, Germany
Andrea Schaerf	Università di Udine, Italy
Marc Sevaux	Université de Bretagne-Sud, France
Kenneth Sörensen	Universiteit Antwerpen, Belgium
Thomas Stützle	Université Libre de Bruxelles, Belgium
Dirk Sudholt	Universität Dortmund, Germany
El-Ghazali Talbi	École Polytechnique Universitaire de Lille, France

Sponsoring Institutions

Universitat Politécnica de Catalunya, Spain
Deutsche Forschungsgemeinschaft, Germany

Table of Contents

Evolutionary Local Search for the Super-Peer Selection Problem and the p-Hub Median Problem

Steffen Wolf and Peter Merz

Distributed Algorithms Group
University of Kaiserslautern, Germany
{pmerz,wolf}@informatik.uni-kl.de

Abstract. Scalability constitutes a key property in Peer-to-Peer environments. One way to foster this property is the introduction of super-peers, a concept which has gained widespread acceptance in recent years. However, the problem of finding the set of super-peers that minimizes the total communication cost is NP-hard. We present a new heuristic based on Evolutionary Techniques and Local Search to solve this problem. Using actual Internet distance measurements, we demonstrate the savings in total communication cost attainable by such a super-peer topology. Our heuristic can also be applied to the more general Uncapacitated Single Assignment p-Hub Median Problem. The Local Search is then further enhanced by generalized *don't look bits*. We show that our heuristic is competitive with other heuristics even in this general problem, and present new best solutions for the largest instances in the well known Australia Post data set.

1 Introduction

During recent years *evolutionary algorithms* enhanced with *local search* have been used to solve many NP-hard optimization problems [1,2,3,4]. These heuristics take their power from the problem specific local search, while keeping all favorable features of the evolutionary approach.

We are especially interested in optimization problems connected with topology construction in Peer-to-Peer (P2P) systems. Well-known properties of these fully decentralized P2P systems include self-organizing and fault-tolerant behavior. In contrast to centralized systems, they usually possess neither a single point of failure nor other bottlenecks that affect the entire network at once. However, the scalability of such networks becomes an issue in the case of excessive growth: Communication times tend to increase and the load put on every node grows heavily when the networks get larger. A possible solution to this issue is the introduction of *super-peers*. Super-peers are peers that act as servers for a number of attached common peers, while at the same time, they form a network of equals among themselves. In a super-peer enhanced P2P network, each common peer is attached to exactly one super-peer, which constitutes its link to the remainder of the network. All traffic will be routed via the super-peers [5,6].

T. Bartz-Beielstein et al. (Eds.): HM 2007, LNCS 4771, pp. 1–15, 2007.

To ensure smooth operation, the peers generally wish to maintain low-delay connections to the other peers. Hence, minimum communication cost is the aim when designing super-peer P2P networks. In this paper, we present a heuristic combining local search with evolutionary techniques for the Super-Peer Selection Problem (SPSP), i. e. the problem of finding the set of super-peers and the assignment of all other peers that minimizes the total communication cost.

Our special interest lies in the construction of these P2P overlay topologies. However, the problem of selecting the super-peers is strongly related to a hub location problem: the *Uncapacitated Single Assignment p-Hub Median Problem* (USApHMP) [7]. The USApHMP is a well known optimization problem and has received much attention in the last two decades. With minor adjustments, our heuristic can also be used for the USApHMP, which allows the comparison with other algorithms on established standard test cases.

This paper is organized as follows. In Section 2, we provide an overview of related work. In Section 3, we propose our Super-Peer Selection Heuristic. In Section 4, we present results from experiments on real world Internet data for the Super-Peer Selection Problem, as well as on standard test cases for the USApHMP, and compare the results with those of other recently published algorithms. The paper concludes with an outline for future research in Section 5.

2 Related Work

The Super-Peer Selection Problem, as proposed here, has not yet been studied in the literature. However, algorithms designed for the USApHMP can also be used for SPSP. The USApHMP has achieved much attention since it was presented by O'Kelly in [7], along with a set of test cases called CAB. Later, O'Kelly *et al.* also presented means of computing lower bounds for these problems [8]. Exact solutions have been computed by Ernst and Krishnamoorthy for problems with up to 50 nodes in [9]. In this paper, they also introduced a new test set called AP. Ebery presented two more efficient mixed integer linear programs (MILP) for the special case of only 2 or 3 hubs [10], and thus solved a problem with 200 nodes (2 hubs), and a problem with 140 nodes (3 hubs). Also, the authors of [9] presented a Simulated Annealing heuristic that found good solutions for problems with up to 200 nodes.

Skorin-Kapov *et al.* presented TabuHub [11], a heuristic method based on tabu search. Results were presented only for the smallest problems of the CAB set ($n \leq 25$). Also, neural network approaches have been proposed for the USApHMP. In [12], the memory consumption and the CPU time for these approaches was reduced. However, the neural network approach was again only applied to the smallest problems in the CAB set ($n \leq 15$). Unfortunately, no computation times are given, making comparisons with other heuristics difficult.

The most promising heuristic for the USApHMP so far has been presented by Pérez *et al.* in [13]. It is a hybrid heuristic combining Path Relinking [14] and Variable Neighborhood Search. The heuristic has proven to be very fast with both the CAB and AP sets, faster than any other heuristic. However, it failed to find the optimum in some of the smaller CAB instances and still left room for improvements in the larger instances of the AP set. The local search neighborhoods used in this heuristic differ from the ones used here. Especially, the most expensive neighborhood is missing in [13]. This explains the speed as well as the loss of quality.

Two Genetic Algorithms have been presented by Kratica *et al.* [15]. These GAs are based on different representations and genetic operations. Both feature a mutation operator that favors the assignment of peers to closer super-peers, as well as a sophisticated recombination. The results of the second GA are the best results so far, as they improved the solutions for the larger AP instances found in [13]. However, the approach does not include a local search, and can still be improved. As far as we know, the heuristic we present in this paper is the first heuristic combining evolutionary techniques with local search.

3 Super-Peer Selection

When constructing a communication cost efficient and load balanced P2P topology we strive for a topology in which a subset of the nodes will function as super-peers while the rest of the nodes, henceforth called edge peers, is each assigned to one of the super-peers. Adhering to the established properties of super-peer overlay structures, the super-peers are fully connected among themselves and are able to communicate directly with the edge peers assigned to them and with their fellow super-peers. Essentially, the super-peers are forming the core of the network. The edge peers, however, will need to route any communication via their assigned super-peer. An example of such a super-peer topology is shown in Fig. 1. Using a topology of this kind, the communication between edge peers p_1 and p_{11} is routed via the super-peers c_1 and c_4. A broadcast in such a topology can be efficiently performed by having one super-peer send the broadcast to all other super-peers, which then forward the message to their respective edge peers. To ensure smooth operation and to ease the load on each peer, the number of super-peers should be limited as well as the number of peers connected to a super-peer.

The Super-Peer Selection Problem can be defined as finding the super-peer topology, i. e. the set of super-peers and the assignment of the edge peers to the super-peers, with minimal total communication cost for a given network. In a P2P setting, this cost can be thought of as the total all-pairs end-to-end communication delay.

3.1 Background

The SPSP is NP-hard [16]. It may be cast as a special case of the Hub Location Problem, first formulated by O'Kelly [7] as a Quadratic Integer Program. In the Hub Location Problem, a number of nodes, the so-called hubs, assume hierarchical superiority over common nodes, a property equivalent to the super-peer concept. Basically, given a network $G = (V, E)$ with $n = |V|$ nodes, p nodes are to be selected as hubs. Let x_{ik} be a binary variable denoting that node i

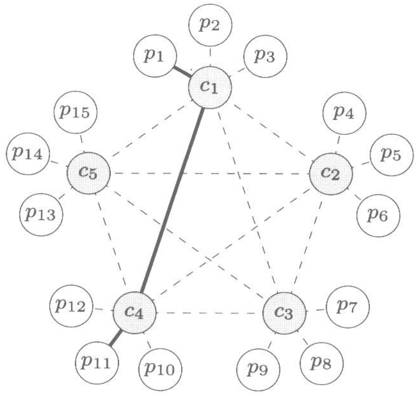


Fig. 1. Example of a P2P network with selected Super-Peers

is assigned to node k if and only if $x_{ik} = 1$. If $x_{kk} = 1$, node k is chosen as a hub. The flow volume between any two nodes $i \neq j$ is equal to one unit of flow. Since all flow is routed over the hubs, the actual weight on the inter-hub links is usually larger than one. The transportation cost of one unit of flow on the direct link between nodes i and j amounts to d_{ij}. Now, the SPSP formulated as a Hub Location Problem is

$$\min Z = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \sum_{k=1}^{n} \sum_{m=1}^{n} (d_{ik} + d_{km} + d_{mj}) \cdot x_{ik} \cdot x_{jm} \tag{1}$$

s. t.

$$x_{ij} \leq x_{jj} \qquad i, j = 1, \ldots, n \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n \tag{3}$$

$$\sum_{j=1}^{n} x_{jj} = p \tag{4}$$

$$x_{ij} \in \{0, 1\} \qquad i, j = 1, \ldots, n \tag{5}$$

Equation (1) yields the total communication cost Z. The set of constraints (2) ensures that nodes are assigned only to hubs, while (3) enforces the allocation of a node to exactly one hub. Due to constraint (4), there will be exactly p hubs.

A more general formulation uses a demand matrix $W = (w_{ij})$. Here, w_{ij} denotes the flow from node i to j in flow units. Also, special discount factors can be applied for the different edge types. Flow between hubs is subject to a discount factor $0 \leq \alpha \leq 1$, flow from a node to its hub is multiplied by a factor δ, and flow from a hub to a common node is multiplied by a factor χ. The total communication cost Z is then:

$$Z = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{m=1}^{n} (\chi \cdot d_{ik} + \alpha \cdot d_{km} + \delta \cdot d_{mj}) \cdot w_{ij} \cdot x_{ik} \cdot x_{jm} \tag{6}$$

The distinction between the different types of edges is motivated by an application in the area of mail transport. Here, the distribution cost differs from the collection cost. Also, the transportation cost between the hubs is assumed to be lower since more efficient means of transport can be used for the accumulated amount of flow. This extension might also be applied in the case of communication networks, especially when asymmetric links are considered. However, the most important difference from the SPSP is the introduction of demand factors w_{ij}, as will be shown in Section 3.3.

Since the objective function in both programs is quadratic and nonconvex, no efficient way to compute the minimum is known. The usual approach is to transform the problem into a Mixed Integer Linear Program (MILP). A straightforward linearization uses $\mathcal{O}(n^4)$ variables. We resort to an MILP formulation using as few as $\mathcal{O}(n^3)$ variables [17]:

$$\min Z = \sum_{i=1}^{n}\sum_{k=1}^{n} (\chi \cdot O_i + \delta \cdot D_i) \cdot d_{ik} \cdot x_{ik} + \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} \alpha \cdot d_{kl} \cdot y_{ikl} \tag{7}$$

s. t. (2), (3), (4), (5),

$$\sum_{l=1}^{n} (y_{ikl} - y_{ilk}) = O_i \cdot x_{ik} - \sum_{j=1}^{n} w_{ij} \cdot x_{jk} \qquad i, k = 1, \ldots, n \tag{8}$$

$$y_{ikl} \geq 0 \qquad i, k, l = 1, \ldots, n \tag{9}$$

Here, $O_i = \sum_{j=1}^{n} w_{ij}$ is the outgoing flow for node i and $D_j = \sum_{i=1}^{n} w_{ij}$ is the demand of node j. Both values can be calculated directly from the problem instance. The variables y_{ikl} denote the flow volume from hub k to hub l which has originated at peer i. Constraints (8) and (9) ensure flow conservation at each node.

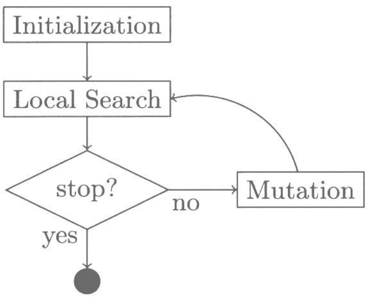
An MILP formulation for the SPSP can be derived by fixing $\chi = \delta = \alpha = 1$, $w_{ij} = 1$ for $i \neq j$, $w_{ii} = 0$, and thus $O_i = D_i = n - 1$:

$$\min Z = \sum_{i=1}^{n}\sum_{k=1}^{n} 2 \cdot (n-1) \cdot d_{ik} \cdot x_{ik} + \sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} d_{kl} \cdot y_{ikl} \tag{10}$$

s. t. (2), (3), (4), (5), (9),

$$\sum_{l=1}^{n} (y_{ikl} - y_{ilk}) = (n-1) \cdot x_{ik} - \sum_{j=1, j \neq i}^{n} x_{jk} \qquad i, k = 1, \ldots, n \tag{11}$$

The factor $2 \cdot (n-1)$ for the edge-peer to super-peer links in (10) is the number of connections using this link. It is based on the assumption that every edge peer needs to communicate with all other $n-1$ peers, and all other peers need to communicate with this edge peer.




```
s ← INITIALIZATION(p)
s ← LOCALSEARCH(s)
β ← n
while ¬ stopping criterion do
  for i = 1 ... m do
    s_i ← MUTATION(β, s)
    s_i ← LOCALSEARCH(s_i)
  end for
  min = argmin_i {Z(s_i)}
  if Z(s_min) < Z(s) then
    s ← s_min
  else
    β ← max{0.8 · β, 2}
  end if
end while
```

Fig. 2. General overview of the Super-Peer Selection Heuristic

These formulations are equivalent to the quadratic formulation only if the distances d_{ij} observe the triangle inequality. Otherwise, the model will generate solutions featuring the property that messages are sent along shortest paths between two hub instead of the intended direct link. The model can still be used for such networks. However, the resulting value can only serve as a lower bound.

The formulation above enables the exact solution of moderately-sized problems (up to 50 peers) in reasonable time, and additionally, the computation of lower bounds for larger networks (up to 150 peers) using its LP relaxation. For networks larger than the given threshold, we use the lower bounds described in [8]. Finally, the sum of all shortest paths' weights yields another lower bound.

3.2 Super-Peer Selection Heuristic

The Super-Peer Selection Heuristic presented here is based on *evolutionary algorithms* and *local search*. It operates on a global view of the network. The general work flow is shown in Fig. 2. The heuristic is quite similar to *iterated local search* [18], but uses more than one offspring solution in each generation.

Representation. A solution is represented by the assignment vector s. For each peer i the value $s(i)$ represents the super-peer of i: $x_{i,s(i)} = 1$. All super-peers, the set of which will be denoted by C, are assumed to be assigned to themselves, i. e. $\forall i \in C : s(i) = i$. For the sake of swift computation, we also store the current capacities of the super-peers, i. e. the number of peers connected to the super-peer: $|V_k| = |\{i \in V \mid s(i) = k\}|$. This set also includes the super-peer itself: $k \in V_k$. The sets V_k are not stored explicitly, but are defined by the assignment vector s.

Initialization. The initial solution is created by randomly selecting p peers as super-peers, and assigning all remaining peers to the nearest super-peer. When handling problems with missing links, this procedure is repeated if the initial set of super-peers is not fully connected.

Local Search. After each step of the Evolutionary Algorithm a local search is applied to further improve the current solution. We use three different neighborhoods: *replacing the super-peer*, *swapping two peers* and *reassigning a peer to another super-peer*. If a neighborhood does not yield an improvement, the next neighborhood is used.

In the first neighborhood the local search tries to replace a super-peer by one of its children. The former child becomes the new super-peer and every other peer that was connected to the old super-peer is reconnected to the new super-peer. The gain of such a move can be computed in $\mathcal{O}(n)$ time. The following formula gives the gain for replacing super-peer k with i:

$$G_{\text{replace}}(i,k) = \sum_{j \in C} 2 \cdot |V_k| \cdot |V_j| \cdot (d_{kj} - d_{ij}) + \sum_{j \in V_k} 2 \cdot (n-1) \cdot (d_{kj} - d_{ij}) \quad (12)$$

If the gain of this move is $G_{\text{replace}}(i,k) > 0$, the move is applied.

The second neighborhood tries to exchange the assignment of two peers. The gain of such a move can be computed in $\mathcal{O}(1)$ time. Since all peers connected to other super-peers are considered as the exchange partner, the total time complexity for searching this neighborhood is $\mathcal{O}(n)$. Using the same notation as before, the following formula gives the gain for swapping the assignments of peers i and j:

$$G_{\text{swap}}(i,j) = 2 \cdot (n-1) \cdot (d_{i,s(i)} + d_{j,s(j)} - d_{i,s(j)} - d_{j,s(i)}) \quad (13)$$

The move with the highest gain is applied if its gain is $G_{\text{swap}}(i,j) > 0$.

The third neighborhood covers the reassignment of a peer to another super-peer. Here, it is important that the capacity limits of the involved super-peers are observed. The gain of reassigning peer i to super-peer k can be calculated in $\mathcal{O}(p)$ time:

$$\begin{aligned} G_{\text{reassign}}(i,k) = \; & 2 \cdot (n-1) \cdot (d_{i,s(i)} - d_{i,k}) \\ & + 2 \cdot \big(|V_{s(i)}| \cdot |V_k| - (|V_{s(i)}| - 1) \cdot (|V_k| + 1)\big) \cdot d_{s(i),k} \\ & + 2 \cdot \sum_{j \in C \setminus \{k, s(i)\}} |V_j| \cdot (d_{s(i),j} - d_{k,j}) \end{aligned} \quad (14)$$

The first part of this equation gives the gain on the link between peer i and its super-peer. The second part gives the gain on the link between the old and the new super-peer. The third part gives the gain on all remaining intra-core links. Out of all super-peers only the one with the highest gain is chosen, thus yielding a total time complexity of $\mathcal{O}(p^2)$. The move is applied only if the total gain is $G_{\text{reassign}}(i,k) > 0$.

These local search steps are performed for each peer i. The local search is restarted whenever an improving step was found and applied. The local search is thus repeated until no improvement for any peer i can be found, i. e. a local optimum has been reached. Since all peers $i \in V$ are considered in these moves, the time complexity for searching the whole neighborhood is $\mathcal{O}(n^2)$.

Mutation. Since local search alone will get stuck in local optima, we use mutation to continue the search. Mutation is done by swapping two random peers. Again, the "gain" of such a swap can be computed by (13). Several mutation steps are applied in each round. The number of mutations is adapted to the success rate. The algorithm starts with $\beta = n$ mutations. If no better solution is found in one generation, the mutation rate β is reduced by 20 %. In each round at least two mutations are applied. This way, the algorithm can adapt to the best mutation rate for the individual problem and for the phase of the search. It is our experience that it is favorable to search the whole search space in the beginning, but narrow the search over time, thus gradually shifting exploration to exploitation.

Population. Our heuristic uses a population of only one individual. There is no need for recombination. This is mainly motivated by the high computation cost and solution quality of the local search. Using mutation and local search, m offspring solutions are created. The best solution is used as the next generation only if it yielded an improvement. This follows a $(1 + m)$ selection paradigm. If there was no improvement in the m children, the mutation rate β is reduced as described before.

Stopping criterion. The heuristic is stopped after five consecutive generations without an improvement. This value is a compromise between solution quality and computation time. In the smaller instances the heuristic often finds the optimum in the first or second generation. Continuing the search would mean to waste time. We also stopped the heuristic after 40 generations regardless of recent improvements. Both values were chosen based on preliminary experiments.

3.3 Adaptation for the USApHMP

The USApHMP introduces weights w_{ij} on the connections between the nodes. While these weights have been equal for all node pairs in the Super-Peer Selection Problem, this is no longer the case in the full USApHMP. The main effect on the heuristic is that we can no longer summarize the flow on the inter-hub edges as $2 \cdot |V_a| \cdot |V_b|$. The following sum has to be used, instead: $\sum_{i \in V_a} \sum_{j \in V_b} w_{ij} + w_{ji}$. This would change the time complexity for calculating the cost of an inter-hub edge from $\mathcal{O}(1)$ to $\mathcal{O}(n^2)$. With the use of efficient data structures, however, the calculation for the cost of a move can be achieved in $\mathcal{O}(n)$ time.

Data structures. In addition to the super-peers' capacities we also store the weights on the p^2 inter-hub links. $WC(a, b) = \sum_{i \in V_a} \sum_{j \in V_b} w_{ij}$ denotes the weight on the link from super-peer a to super-peer b. In each move made by the local search or the mutation these weights are changed accordingly. Only the selection of a new super-peer does not change these weights. Also, the gain calculations have to be adapted:

$$G_{\text{replace}}(i, k) = \alpha \cdot \sum_{j \in C} (WC(j, k) + WC(k, j)) \cdot (d_{kj} - d_{ij}) + \sum_{j \in V_k} (\chi \cdot O_j + \delta \cdot D_j) \cdot (d_{kj} - d_{ij}) \tag{15}$$