Hybrid Bridge-Based Memetic Algorithms for Finding Bottlenecks in Complex Networks

David Chalupa[a,b], Ken A. Hawick[b], James A. Walker[b]

[a]*Operations Research Group*
Department of Materials and Production
Aalborg University
Fibigerstræde 16, Aalborg 9220, Denmark
Email: `dc@m-tech.aau.dk`
[b]*School of Engineering and Computer Science*
University of Hull
Cottingham Road, Hull HU6 7RX, UK.
Email: {`k.a.hawick,j.a.walker`}`@hull.ac.uk`

Abstract

We propose a memetic approach to find bottlenecks in complex networks based on searching for a graph partitioning with minimum conductance. Finding the optimum of this problem, also known in statistical mechanics as the Cheeger constant, is one of the most interesting NP-hard network optimisation problems. The existence of low conductance minima indicates bottlenecks in complex networks. However, the problem has not yet been explored in depth in the context of applied discrete optimisation and evolutionary approaches to solve it. In this paper, the use of a memetic framework is explored to solve the minimum condutance problem. The approach combines a hybrid method of initial population generation based on bridge identification and local optima sampling with a steady-state evolutionary process with two local search subroutines. These two local search subroutines have complementary qualities. Efficiency of three crossover operators is explored, namely one-point crossover, uniform crossover, and our own partition crossover. Experimental results are presented for both artificial and real-world complex networks. Results for Barabási-Albert model of scale-free networks are presented, as well as results for samples of social networks and protein-protein interaction networks. These indicate that both well-informed initial population generation and the use of a crossover seem beneficial in solving the problem in large-scale.

Preprint submitted to Big Data Research *January 24, 2018*

Keywords:
memetic algorithms, bottlenecks, complex networks, minimum conductance problem, sparsest cut, Cheeger constant

1. Introduction

Analysis of complex networks has motivated the study of a variety of hard computational problems. The search for bottlenecks has become one of the wider computational problems in complex networks, motivated by a number of applications in social networks [1, 2, 3], biological networks [4, 5], power grids [6, 7] or water distribution networks [8]. While these problems are diverse in their applications, many of them can be transformed to a number of metrics [9] and algorithmic tools to explore local and global robustness of network structure.

One of the most popular metrics indicating the existence of a bottleneck in a complex network is its conductance [10]. Informally, minimum conductance of a network is the minimum ratio of the number of edges connecting two disjoint partitions of its vertices, and the minimum number of edges incident to vertices of one of these partitions. Conductance is a value between 0 and 1, with 0 indicating that the network is disconnected, while 1 indicates that the network is fully connected. Usually, a complex network has a bottleneck if its minimum conductance is a small value close to 0.

The application areas for bottleneck identification and conductance optimisation are wide and include algorithms for exploration of protein-protein interactions [11], community detection [2, 12, 13, 14], understanding of group formation [15], data mining in social media [16], cyberattack detection [17], or congestion reduction in transportation networks [18].

In the literature, the minimum conductance problem has also been called the sparsest cut problem [19]. In statistical mechanics and mathematics, the minimum value of conductance in a complex network is often referred to as its Cheeger constant [20]. The minimum conductance problem is known to be NP-hard for over a decade [21]. It has also been shown that for general graphs, it is intractable to approximate the minimum conductance within any constant factor [22].

Contributions. In the present paper we propose a new bridge-based memetic approach to finding low-conductance partitions of complex networks, representing their bottlenecks.

A framework of a specialised steady-state adaptive memetic algorithm (StS AMA) is presented to solve the problem. Several of its variants are explored, including variants with three different crossover operators, as well as a crossover-free variant of population-based local search (PBLS). The algorithms start with an initial population generated with an adaptive probability of 1-bits in the initial solution. This allows the approach to explore regions of the search space with highly imbalanced partitions. Our preliminary investigations uncovered that these are difficult to reach by more conventional evolutionary approaches, especially if there are relatively balanced partitions that serve as strong search space attractors [23].

Several evolutionary approaches are explored, including PBLS, StS AMA with one-point crossover (1PX), uniform crossover (UX) and our own partition crossover (PartX). Each of these is applied both in its plain variant and a bridge-based variant (PBLS-B and StS AMA-B). The bridge-based variants use Tarjan's bridge identification algorithm [24] to generate a promising partitioning that is put into the initial population. All of the algorithms use two local search subroutines. Randomised local search $\mathrm{RLS}^{1,2}$ allows moves of single vertices between partitions, as well as swaps. Local search LS^{1} is a best improvement algorithm only allowing moves of single vertices, always leading to a local optimum. This ensures that PBLS and StS AMA operate with a population of local optima at all times during the evolutionary process.

The experimental results are presented for a selection of social network samples, protein-protein interaction networks, as well as several graphs from network science literature. Confronting PBLS and StS AMA, we found that crossover operators seem beneficial in solving the minimum conductance problem. However, our results also show that different crossover operators tend to work for different problem instances. For some networks, the bridge-based approach has also been highly successful, while it had little effect for other networks. The bottlenecks found are also very interesting from the application perspective. While for some instances, the bottlenecks were identified in the form of a relatively small cluster that is sparsely connected to the rest of the network, some high-quality solutions found are quite balanced.

2. Background and Related Work

Let $G = [V, E]$ be an undirected graph and let $S \subseteq V$. Then, conductance of the partitioning of V into sets S and $V \backslash S$ is defined by:

$$\Phi(S) = \frac{c_G(S)}{\min\{Vol(S), Vol(V \backslash S)\}}, \tag{1}$$

where $c_G(S) = \{\{v, w\} : \ v \in S \ \wedge \ w \in V \backslash S\}$ is the number of edges connecting the two sets S and $V \backslash S$ and

$$Vol(S) = \sum_{v \in S} deg(v) \tag{2}$$

is the volume of subset S. In other words, conductance determines the ratio of the number of edges connecting the sets S and $V \backslash S$ to the total number of edges incident to the partition with lower volume. We will refer to the problem of finding $S \subseteq V$ such that $\Phi(S)$ is minimised as the minimum conductance problem.

To be more specific, this conductance will be referred to as the symmetric conductance. One can see that $\Phi(S) = \Phi(V \backslash S)$, i.e. reversing S and $V \backslash S$ does not have an impact on conductance. This is also the optimisation problem, for which the optimum is called Cheeger constant of the network [20]. NP-hardness has been proven for the symmetric variant of minimum conductance problem [21].

It is worth noting that in a number of studies, an alternative definition of asymmetric conductance is also used [25]:

$$\Phi_a(S) = \frac{c_G(S)}{Vol(S)}. \tag{3}$$

This alternative definition of conductance leads to the same value as the symmetric one if $Vol(S) \leq Vol(V \backslash S)$. For its simplicity, it is often used as a metric for evaluation of community detection algorithms [2, 12]. However, it can have a different optimum than the symmetric variant that does not have to be equal to the Cheeger constant.

As indicated above, conductance has been widely used as a metric for evaluation of community detection algorithms. The vast amounts of large-scale real-world complex network data have motivated development of a variety of algorithmic approaches to community detection [3, 26, 27]. Leskovec et al. have developed the network community profile concept [2, 12] that represents a function mapping the community size value to the conductance of the best community found by a community detection algorithm. This represents a size-dependent view on the minimum conductance problem, in which the value of conductance for a particular partitioning is measured as a function of community size.

Van Laarhoven and Marchiori have explored a continuous generalisation of the asymmetric conductance variant for local community detection and its optimisation using gradient descent and expectation minimisation algorithms [25]. Their approach has been shown to be highly scalable, providing solutions with very good conductance compared to more conventional community detection algorithms. To the best of our knowledge, this is the first study tackling a variant of a conductance problem empirically as an optimisation problem. This study was focused on the asymmetric variant of the problem and used a transformation of the problem to the continuous domain. However, it seems that neither the symmetric variant has been studied in experimental literature, nor it has been tackled using discrete optimisation techniques. Such studies seem to be of a high interest, since the symmetric variant is known to be NP-hard [21].

As a typical 0-1 optimisation problem, the minimum conductance problem can be formulated as a pseudo-Boolean function. For some pseudo-Boolean optimisation problems such as max-SAT or NK landscapes [28], it is possible to use efficient partition crossover operators [29] and very efficient local search strategies [30] using constant-time steepest descent [31]. However, it remains open whether this is possible for the minimum conductance problem and it can also be influenced by whether symmetric or asymmetric variant of the problem is studied. Preliminary empirical evidence suggests that randomised population-based search and adaptive strategies work much better than simple steepest descent for the symmetric variant [23].

3. Bridge-based Memetic Algorithms for Finding Bottlenecks in Complex Networks

In this section our memetic approach is introduced to solve the minimum conductance problem, identifying bottlenecks in complex networks. Memetic algorithms have been successfully applied to a number of community detection [32] and graph partitioning problems [33]. One can therefore expect this approach to be promising also for minimisation of conductance.

Algorithm 1: Steady-state Adaptive Memetic Algorithm (StS AMA) for the Minimum Conductance Problem

Input: population size p, tournament size t,
local search length l, crossover type $X_t \in \{1PX, UX, PartX\}$
Output: best configuration P_{best} found

1 initialise population P with p individuals
2 while stopping criteria are not met
3   pick parents P_{p_1} and P_{p_2} such that $p_1 \neq p_2$ using
      a tournament of size t
4   create the offspring O using crossover of type X_t
5   improve the offspring O using $\text{RLS}^{1,2}$ for l iterations
6   improve the offspring O using LS^1 until the local optimum is reached
7   if $O \notin P$ then replace the worst individual in P with O
8 return the best individual P_{best} in P

The general framework of the algorithms is described first. Next, we focus on the initial population generation and the bridge-based component of our approach. This is then followed by a description of the crossover operators used, as well as the two local search subroutines of the memetic approach.

3.1. General Framework

The general idea of our approach is based on a steady-state evolutionary algorithm framework [34]. In the following, the main focus will be on the design of StS AMA, as our core algorithmic approach to solve the problem. StS AMA-B differs from StS AMA only in the use of bridge-based component in the initial solution generation. PBLS is also very similar to StS AMA and differs only by in use of cloning of a single parent, rather than a crossover of two parents. Otherwise, all algorithms studied are based on the same general framework.

The pseudocode of StS AMA is presented in Algorithm 1. In step 1, the initial population is generated. The details of this process will be given in Algorithm 2. This is followed by an evolutionary process. In steps 3-4, parents P_{p_1} and P_{p_2} are chosen and a single offspring O is created from them by the use of crossover operator of type X_t. Note that X_t is an input of the algorithm, leading to several of its variants studied. In steps 5-6, O is first improved by randomised local search algorithm $\text{RLS}^{1,2}$ for a predefined number of iterations. O is then improved using local search algorithm LS^1

Algorithm 2: Initial population generation for StS AMA-B

Input: population size p
Output: initial population P

1 find the set of all bridges B using Tarjan's algorithm
2 let P_1 be a partitioning such that each bit of P_1 is 0
3 for $b \in B$
4   construct candidate partitioning P_c of the network around b
    as shown in Figure 1
5   if $\Phi(P_1) \geq \Phi(P_c)$
6         $P_1 = P_c$
7 for $i = 2...p$
8     $p_s = 1/2$
9   do
10   set each bit of a candidate for individual P_i to 1 with probability p_s
11   improve the candidate for individual P_i using LS[1]
        until the local optimum is reached
12         $p_s = p_s/2$
13   while the current candidate for P_i is at least as good
    as the best of the previous candidates
14   set the best candidate sampled in steps 9-13 as the individual P_i
15 return $P = \{P_1, P_2, ..., P_p\}$

until the algorithm makes sure that O is a local optimum. In step 7, the worst individual in the population is simply replaced with O.

3.2. Initial Population Generation and Bridge Identification

The exact details of initial population generation depend on whether StS AMA or StS AMA-B is used. For StS AMA-B, this procedure combines the idea of an adaptive probability of 1-bit generation, as well as bridge identification to construct a potentially promising partitioning that is put into the initial population. Note that in the bit-based representation, a 1-bit represents that a vertex is in S and a 0-bit represents that it is in $V \backslash S$ (or vice versa, as the problem is symmetric). These ideas are used to ensure that the initial solutions are of a good quality and take the specific properties of our problem into account.

Algorithm 2 presents the pseudocode of initial population generation for StS AMA-B. For StS AMA, this procedure differs by not using the bridge

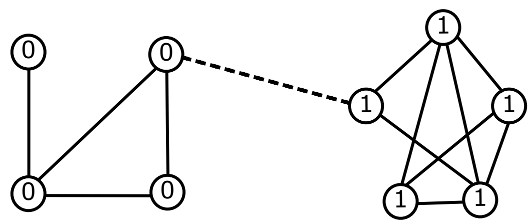

Figure 1: An illustration of a bridge (depicted by a dashed line) as a bottleneck in a graph with a pronounced clustered structure. The bridge can be identified in polynomial time, followed by labelling of the two partitions, establishing a potentially promising initial solution for the evolutionary process.

identification. Steps 1-6 are skipped in StS AMA and all p initial solutions are generated using steps 7-13.

In step 1, Tarjan's algorithm for finding all bridges in the graph is used [24], based on the framework of depth-first search. In steps 2-6, individual P_1 is generated. Step 2 simply initialises P_1 with a solution with an infinitely high conductance. In steps 3-6, the algorithm scans all partitions induced by all bridges identified. For a bridge b, S and $V \backslash S$ represent partitions that are connected by b only, as shown in Figure 1. Individual P_1 will then simply be the partition with minimum conductance.

In steps 7-13, the rest of the population is generated. This process facilitates variable p_s, which denotes the probability that a 1-bit is generated. With this probability, a candidate solution is generated. In the beginning, a 1-bit is generated with the same probability as a 0-bit, leading to balanced partitions. Such a candidate solution is improved to a local optimum using local search algorithm LS[1] described below. The value of p_s is then halved and the process is repeated. If the next solution generated is of a better quality, then p_s is halved again. This is iterated until a worsening in the quality of the solution sampled is obtained. This ensures that highly imbalanced partitions are also explored.

It is worth noting that although it is possible to find all bridges in polynomial time, not every graph however has bridges that would make "good" bottlenecks. A typical case of such bottlenecks is represented by edges adjacent to leaves. This is why StS AMA uses the hybrid initialisation procedure above to sample an initial population of potentially imbalanced promising bottlenecks. In the experimental results below, it will be demonstrated that the qualities of bottlenecks identified heavily depend on structural and quantitative properties of the network analysed.

To summarise the general framework, as well as the initial solution generation procedure, the flowchart representing StS AMA is given in Figure 2.

3.3. Crossover Operators

Three variants of StS AMA will be studied, that utilise three different crossover operators. Each of the variants will be studied separately, to determine which crossover operators work for the problem and what is the impact of crossover on the efficiency of StS AMA in general.

One-point crossover (1PX). This operator simply takes two parents P_1 and P_2 and chooses the crossover point t uniformly at random. Bits with indices 1, 2, ..., $t-1$ are then taken from P_1 and the rest of the bits are taken from P_2. Only one offspring O is created this way, to easily compare this strategy to the other two crossover operators, which produce only one offspring.

Uniform crossover (UX). The uniform crossover treats each bit separately and takes it from parent P_1 with probability $1/2$ and from parent P_2 otherwise.

Partition crossover (PartX). We have also designed and experimented with a simple partition crossover for the problem. Let parent P_1 consist of partitions S_1 and $V \backslash S_1$ and P_2 consist of partitions S_2 and $V \backslash S_2$. Let $s(A, B) = |A \cap B|$ be the similarity of sets A and B. The partition crossover computes similarities $s(S_1, S_2)$, $s(S_1, V \backslash S_2)$, $s(S_2, V \backslash S_1)$ and $s(V \backslash S_1, V \backslash S_2)$. The highest similarity is then taken and the intersection of the two corresponding sets is put into the partition S of the offspring O. The vertices of this partition are then excluded and similarities are recalculated. The intersection with the highest similarity after this update is then put into the partition $V \backslash S$ of the offspring O. The remaining vertices are assigned into S or $V \backslash S$ uniformly at random.

3.4. Local Search Strategies

Our approach uses two local search strategies to improve candidate solutions and ensure that all members of the population represent local optima at all times. Both of these local search subroutines can implemented efficiently if the objective function is recalculated after a single bit flip in $\mathcal{O}(1)$ time using auxiliary data [23].

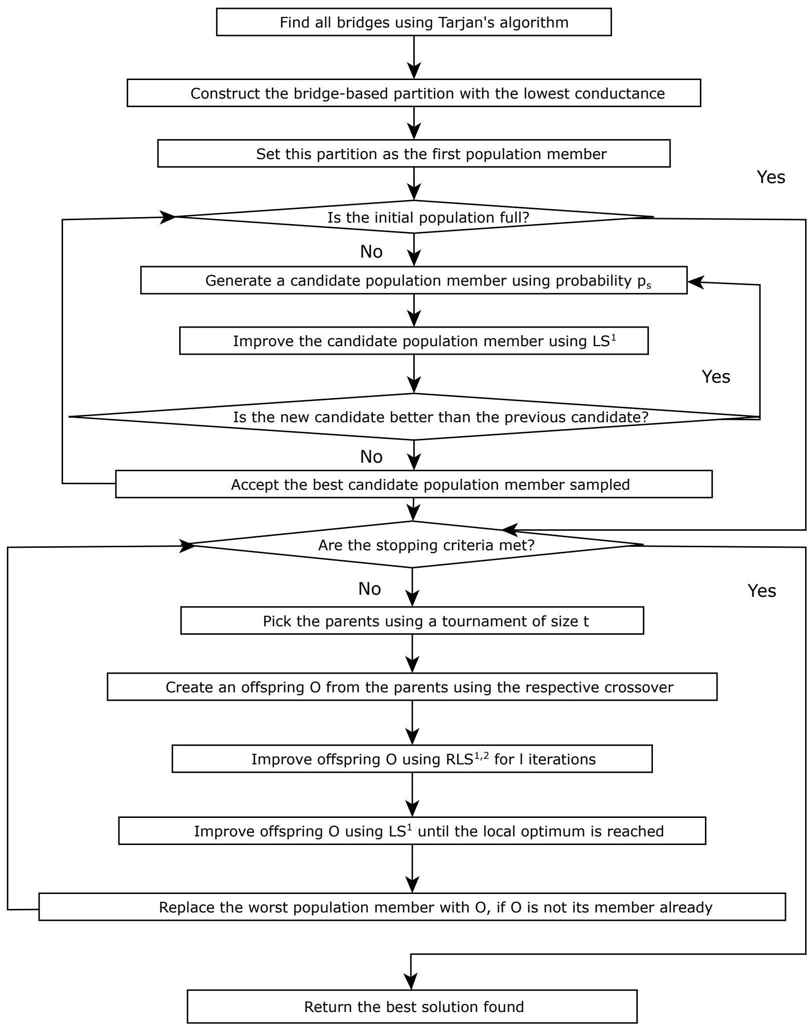

Figure 2: A flowchart representing the workings of StS AMA-B, including the bridge-based component, the initial population generation, as well as the evolutionary process.

Let $S \subseteq V$ represent the current solution and let $S' = S \cup \{v\}$, i.e. S' will be the solution obtained by moving v from partition $V \backslash S$ into partition S. We will then have that:

$$c_G(S') = c_G(S) - deg_S(v) + deg_{V \backslash S}(v), \tag{4}$$

where $deg_S(v)$ is the number of neighbours of v that are in partition S. The volumes of new partitions can also be recalculated as follows:

$$Vol(S') = Vol(S) + deg(v), \tag{5}$$

$$Vol(V \backslash S') = Vol(V \backslash S) - deg(v). \tag{6}$$

This implies that $\Phi(S')$ can be recalculated from $\Phi(S)$ in $\mathcal{O}(1)$ time if the current values of $deg_S(v)$ and $deg_{V \backslash S}(v)$ are stored in auxiliary arrays for each vertex v.

After a move modifying S to S' is accepted, values of $deg_S(w)$ and $deg_{V \backslash S}(w)$ can be updated using the following rules, for all neighbours w of v, i.e. $w \in V$ such that $\{v, w\} \in E$:

$$deg_{S'}(w) = deg_S(w) + 1, \tag{7}$$

$$deg_{V \backslash S'}(w) = deg_{V \backslash S}(w) - 1. \tag{8}$$

Randomised local search RLS^{1,2}*.* At each time step, randomised local search attempts to flip either one or two bits. One bit is flipped with probability 1/2, two bits are flipped otherwise. A flip of a single randomly chosen bit effectively represents a move of the corresponding vertex from S into $V \backslash S$ or vice versa. A flip of two bits also allows the algorithms to potentially perform two moves at once, including swaps. A subroutine of RLS^{1,2} is stopped after l iterations.

Local search LS^{1}*.* This algorithm attempts to flip each bit separately and chooses the best of these moves. If none of these moves lead to an improvement, then the current solution S represents a local optimum. A subroutine of LS^{1} stops whenever such a local optimum is reached.

Table 1: An overview of the identifiers used for the eight algorithms studied.

	without bridge identification	with bridge identification
no crossover	PBLS	PBLS-B
one-point crossover	StS AMA 1PX	StS AMA 1PX-B
uniform crossover	StS AMA UX	StS AMA UX-B
partition crossover	StS AMA PartX	StS AMA PartX-B

Table 2: An overview of the parameter values used in our experiments.

parameter		value
population size		$\lvert P \rvert = 100$
tournament size		$t = 2$
local search length l for $RLS^{1,2}$		$l = 10^6$
maximum number of generations	short runs	500
	long runs	10000

4. Experimental Results

In this section, the experimental results of StS AMA and its variants are presented. We will first discuss the experimental design and problem instances used. Next, the results obtained for synthetic scale-free networks of different sizes will be presented. This will be followed by the results obtained for real-world network data. Last but not least, we provide a brief discussion of our findings and implications for future research.

4.1. Experimental Design

Eight algorithm variants have been computationally studied. All algorithms follow the general framework of StS AMA specified above. The identifiers of all eight algorithms studied are given in Table 1. PBLS represents the population-based local search variant without crossover, using only solution cloning and local search. PBLS is included to provide a comparison of StS AMA to an equivalent crossover-free algorithm, investigating the usefulness and efficiency of crossover operators. StS AMA 1PX uses one-point crossover, StS AMA UX uses uniform crossover and StS AMA PartX uses the partition crossover. PBLS-B, StS AMA-B 1PX, StS AMA-B UX and StS AMA-B PartX represent the bridge-based variants of the algorithms, i.e. the variants where one of solutions in the initial population is generated using a bridge-based partitioning.

Parameter values used in our experiments are summarised in Table 2. All algorithms were used with a population of 100 individuals, tournament size $t = 2$ and $\text{RLS}^{1,2}$ with local search length $l = 10^6$. Each experiment was terminated if a maximum number of generations was reached, aiming for a platform-independent study of the techniques. For synthetic networks, we ran each of the algorithms on 100 independently generated networks for a maximum of 500 generations. The average performance was then compared. For real-world networks, short-running experiments using 100 runs were first performed with a maximum of 500 generations. A subset of the most promising algorithms was then selected also to perform 30 long runs with a maximum of 10000 generations.

The algorithms were run on real-world network data from several sources, mainly focused on social and biological networks. Social network data used includes samples of different sizes from Google+, as well as social network Pokec that has been previously studied in its entirety [35] and its large snapshot is a part of the SNAP network data repository [36]. We also present the results obtained for protein-protein interaction network from the UCLA database of interacting proteins [37, 38, 39, 40]. Results for instances from Newman's network repository have also been used [4, 41, 42, 43, 44]. A large group of the instances used in this study have also been previously used in studies of long cycles [45] and k-reachability in complex networks [46].

Interested reader can also refer to the preliminary work [23], which investigated the performance of $\text{RLS}^{1,2}$, LS^1, their adaptive versions, as well as simple genetic algorithms in solving this problem. In the following, the focus will be on the results obtained by the advanced forms of StS AMA.

4.2. Results for Synthetic Complex Networks

Before embarking on an evaluation of the approaches for real-world data, it was decided to study the algorithms in solving the problem for synthetic networks with variable sizes and properties. We have used the Barabási-Albert model of preferential attachment to generate a sequence of scale-free networks [47, 48]. These networks are characterised by the number of vertices n and the number of incoming edges per vertex w. Their degree distribution

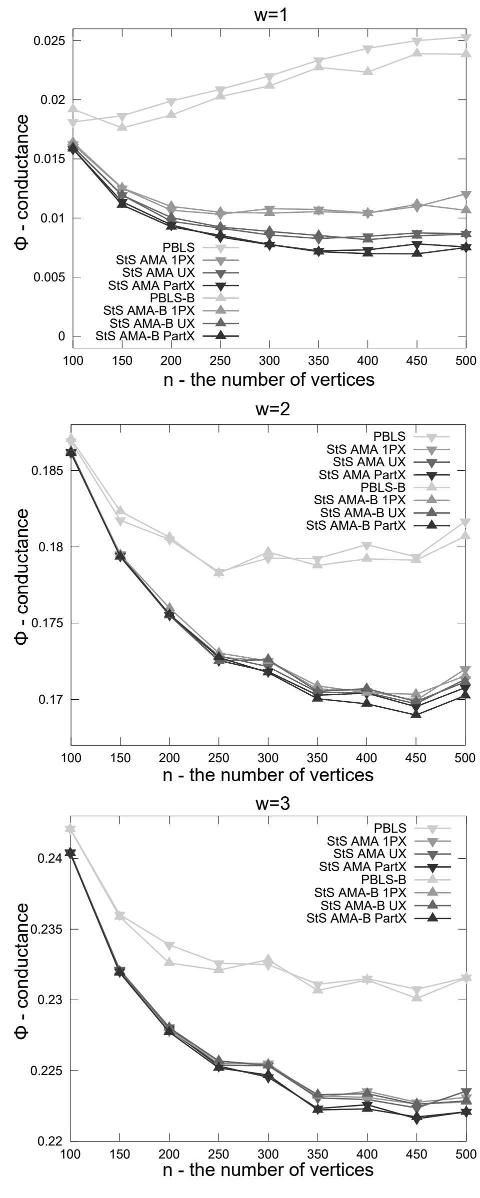

Figure 3: The plots representing the average conductance found by each algorithm for scale-free networks generated by Barabási-Albert model. The networks studied have from 100 to 500 vertices and were generated with $w = 1, 2$ and 3 incoming edges per vertex.

follows a power law, similarly to many social and biological networks. The algorithms were then applied to networks generated with different values of n and w and the performance of the algorithms has been studied.

It is worth noting that in our preliminary experiments, it was possible to use simple exhaustive search to find a proven optimum for the problem in networks with up to between 30-40 vertices. For networks with up to 20 vertices, it is also possible to generate thousands of networks and solve the problem using exhaustive search, estimating a "typical" minimum conductance of small scale-free networks. The largest network for which the problem was solved exactly was generated for $n = 38$ and $w = 2$ and the process took several hours on a standard desktop machine. For the small networks we generated, StS AMA was generally able to find solutions with the same conductance as the exhaustive search. This confirmed that the approach works well for small instances.

To provide a better insight into the performance, the problem was then solved for larger synthetic networks using all eight algorithms described above. We used networks with $n = 100, 150, ..., 500$ vertices and with $w = 1, 2$ and 3 incoming edges per vertex. The results obtained are depicted in Figure 3. One can observe that all variants of StS AMA perform better than PBLS. This indicates that the use of a crossover indeed helps in improving the performance of an evolutionary algorithm, compared to a population-based algorithm based purely on local search. However, these results also do not clearly indicate which of the crossover operators works best. All algorithms with a crossover seem to lead a relatively similar profile of the conductance sampled, with only minor fluctuations. This suggests that crossover plays a role as a diversification operator, rather than an intensification operator. The impact of the bridge-based component of the algorithm also does not seem to be entirely clear. However, this will partially contrast with the results obtained for some real-world networks, for which the bridge-based component played a significant role.

4.3. Results for Real-world Networks

This section presents the results obtained for real-world network data. We first focus on the results obtained mainly for social and protein-protein interaction networks with a maximum of 500 generations. Next, we will present the results of bridge-based algorithms with crossover also in long runs with a maximum of 10000 generations.

Table 3: Experimental results comparing the population-based local search (PBLS) with variants of the steady-state memetic algorithm (StS AMA) with one-point (1PX), uniform (UX) and partition crossovers (PartX) in short runs within a maximum of 500 generations for social networks and adjective-noun adjacency network. Bridge-based variants (-B) of the algorithms are also included in the comparison.

G	algorithm	$\min \Phi(S)$	$E[\Phi(S)]$	success rate
gplus_500	PBLS	**0.02040816**	0.04384623	9 / 100
	StS AMA 1PX	**0.02040816**	0.0333401	11 / 100
	StS AMA UX	**0.02040816**	0.03319711	6 / 100
	StS AMA PartX	**0.02040816**	0.03307758	8 / 100
	PBLS-B	**0.02040816**	**0.02040816**	100 / 100
	StS AMA-B 1PX	**0.02040816**	**0.02040816**	100 / 100
	StS AMA-B UX	**0.02040816**	**0.02040816**	100 / 100
	StS AMA-B PartX	**0.02040816**	**0.02040816**	100 / 100
gplus_2000	PBLS	0.05278922	0.06109376	1 / 100
	StS AMA 1PX	0.04941531	0.05040731	3 / 100
	StS AMA UX	0.0493756	0.04962178	2 / 100
	StS AMA PartX	0.0493756	0.04966333	2 / 100
	PBLS-B	0.05408946	0.06131752	1 / 100
	StS AMA-B 1PX	0.04941531	0.05044017	3 / 100
	StS AMA-B UX	**0.04931558**	0.04960255	1 / 100
	StS AMA-B PartX	0.04937113	**0.04957637**	1 / 100
pokec_2000	PBLS	0.02517306	0.03268988	2 / 100
	StS AMA 1PX	**0.02360775**	0.0251969	8 / 100
	StS AMA UX	0.02470694	0.02490648	7 / 100
	StS AMA PartX	0.02433015	0.02476391	1 / 100
	PBLS-B	0.02643456	0.03242954	1 / 100
	StS AMA-B 1PX	**0.02360775**	0.02523519	3 / 100
	StS AMA-B UX	0.02389706	0.02482624	1 / 100
	StS AMA-B PartX	0.02470694	**0.02475772**	22 / 100
gplus_10000	PBLS	0.0706565	0.07257838	1 / 100
	StS AMA 1PX	0.06729367	0.06803327	1 / 100
	StS AMA UX	0.06602772	0.06672153	1 / 100
	StS AMA PartX	0.06607105	0.06680015	1 / 100
	PBLS-B	**0.04347826**	**0.04347826**	100 / 100
	StS AMA-B 1PX	**0.04347826**	**0.04347826**	100 / 100
	StS AMA-B UX	**0.04347826**	**0.04347826**	100 / 100
	StS AMA-B PartX	**0.04347826**	**0.04347826**	100 / 100
pokec_10000	PBLS	0.03030303	0.05041547	3 / 100
	StS AMA 1PX	0.03030303	**0.04693046**	1 / 100
	StS AMA UX	0.04245283	0.05102121	1 / 100
	StS AMA PartX	0.04500978	0.05075255	1 / 100
	PBLS-B	0.02428256	0.05113928	1 / 100
	StS AMA-B 1PX	0.03030303	0.04714125	2 / 100
	StS AMA-B UX	0.04698492	0.05074653	1 / 100
	StS AMA-B PartX	**0.02083333**	0.05034511	1 / 100
adjnoun [41]	PBLS	**0.27830179**	0.2818578	21 / 100
	StS AMA 1PX	**0.27830179**	0.27856984	85 / 100
	StS AMA UX	**0.27830179**	0.27870569	77 / 100
	StS AMA PartX	**0.27830179**	0.27851513	90 / 100
	PBLS-B	**0.27830179**	0.28255219	11 / 100
	StS AMA-B 1PX	**0.27830179**	0.27868494	80 / 100
	StS AMA-B UX	**0.27830179**	0.27871893	77 / 100
	StS AMA-B PartX	**0.27830179**	**0.27843396**	94 / 100

Table 4: Experimental results comparing the population-based local search (PBLS) with variants of the steady-state memetic algorithm (StS AMA) with one-point (1PX), uniform (UX) and partition crossovers (PartX) in short runs within a maximum of 500 generations for protein-protein interaction networks. Bridge-based variants (-B) of the algorithms are also included in the comparison.

G	algorithm	$\min \Phi(S)$	$E[\Phi(S)]$	success rate
$Celeg$20160114	PBLS	**0.01226994**	0.02426582	31 / 100
	StS AMA 1PX	**0.01226994**	0.02656605	26 / 100
	StS AMA UX	**0.01226994**	0.0309486	29 / 100
	StS AMA PartX	**0.01226994**	0.03162063	34 / 100
	PBLS-B	**0.01226994**	0.02233454	34 / 100
	StS AMA-B 1PX	**0.01226994**	**0.02166496**	38 / 100
	StS AMA-B UX	**0.01226994**	0.02258327	37 / 100
	StS AMA-B PartX	**0.01226994**	0.02799719	28 / 100
$Dmela$20160114	PBLS	0.15517241	0.18857478	1 / 100
	StS AMA 1PX	0.15447154	0.18627825	1 / 100
	StS AMA UX	0.17592593	0.18721519	1 / 100
	StS AMA PartX	0.14685315	0.18738843	1 / 100
	PBLS-B	**0.03030303**	**0.03030303**	100 / 100
	StS AMA-B 1PX	**0.03030303**	**0.03030303**	100 / 100
	StS AMA-B UX	**0.03030303**	**0.03030303**	100 / 100
	StS AMA-B PartX	**0.03030303**	**0.03030303**	100 / 100
$Ecoli$20160114	PBLS	**0.03333333**	0.30560565	3 / 100
	StS AMA 1PX	**0.03333333**	0.30487322	1 / 100
	StS AMA UX	**0.03333333**	0.30546100	1 / 100
	StS AMA PartX	**0.03333333**	0.29887465	2 / 100
	PBLS-B	**0.03333333**	0.06612821	1 / 100
	StS AMA-B 1PX	0.06024096	0.06660241	1 / 100
	StS AMA-B UX	0.06666667	0.06666667	100 / 100
	StS AMA-B PartX	**0.03333333**	**0.06600000**	2 / 100
$Hpylo$20160114	PBLS	0.15594974	0.16692206	1 / 100
	StS AMA 1PX	**0.14855876**	0.15361087	1 / 100
	StS AMA UX	**0.14855876**	0.15290175	1 / 100
	StS AMA PartX	0.14899926	**0.1512717**	1 / 100
	PBLS-B	0.15447154	0.16555901	1 / 100
	StS AMA-B 1PX	0.14962963	0.15426639	1 / 100
	StS AMA-B UX	0.14918759	0.15328013	5 / 100
	StS AMA-B PartX	**0.14855876**	0.15140646	2 / 100
$Hsapi$20160114	PBLS	0.05758139	0.06415486	1 / 100
	StS AMA 1PX	0.04545899	0.04880509	1 / 100
	StS AMA UX	0.03705152	0.04247478	1 / 100
	StS AMA PartX	0.03792766	0.04236963	1 / 100
	PBLS-B	**0.00729927**	**0.00729927**	100 / 100
	StS AMA-B 1PX	**0.00729927**	**0.00729927**	100 / 100
	StS AMA-B UX	**0.00729927**	**0.00729927**	100 / 100
	StS AMA-B PartX	**0.00729927**	**0.00729927**	100 / 100
$Mmusc$20160114	PBLS	0.01438849	0.03632427	1 / 100
	StS AMA 1PX	0.0166158	0.0250686	1 / 100
	StS AMA UX	0.01343183	0.01960684	1 / 100
	StS AMA PartX	0.01276024	0.01582498	1 / 100
	PBLS-B	**0.00578035**	**0.00578035**	100 / 100
	StS AMA-B 1PX	**0.00578035**	**0.00578035**	100 / 100
	StS AMA-B UX	**0.00578035**	**0.00578035**	100 / 100
	StS AMA-B PartX	**0.00578035**	**0.00578035**	100 / 100
$Scere$20160114	PBLS	0.23796338	0.23828424	1 / 100
	StS AMA 1PX	0.23771743	0.23787464	1 / 100
	StS AMA UX	0.20287425	0.23594457	1 / 100
	StS AMA PartX	0.23771998	0.2378532	1 / 100
	PBLS-B	**0.14285714**	**0.14285714**	100 / 100
	StS AMA-B 1PX	**0.14285714**	**0.14285714**	100 / 100
	StS AMA-B UX	**0.14285714**	**0.14285714**	100 / 100
	StS AMA-B PartX	**0.14285714**	**0.14285714**	100 / 100

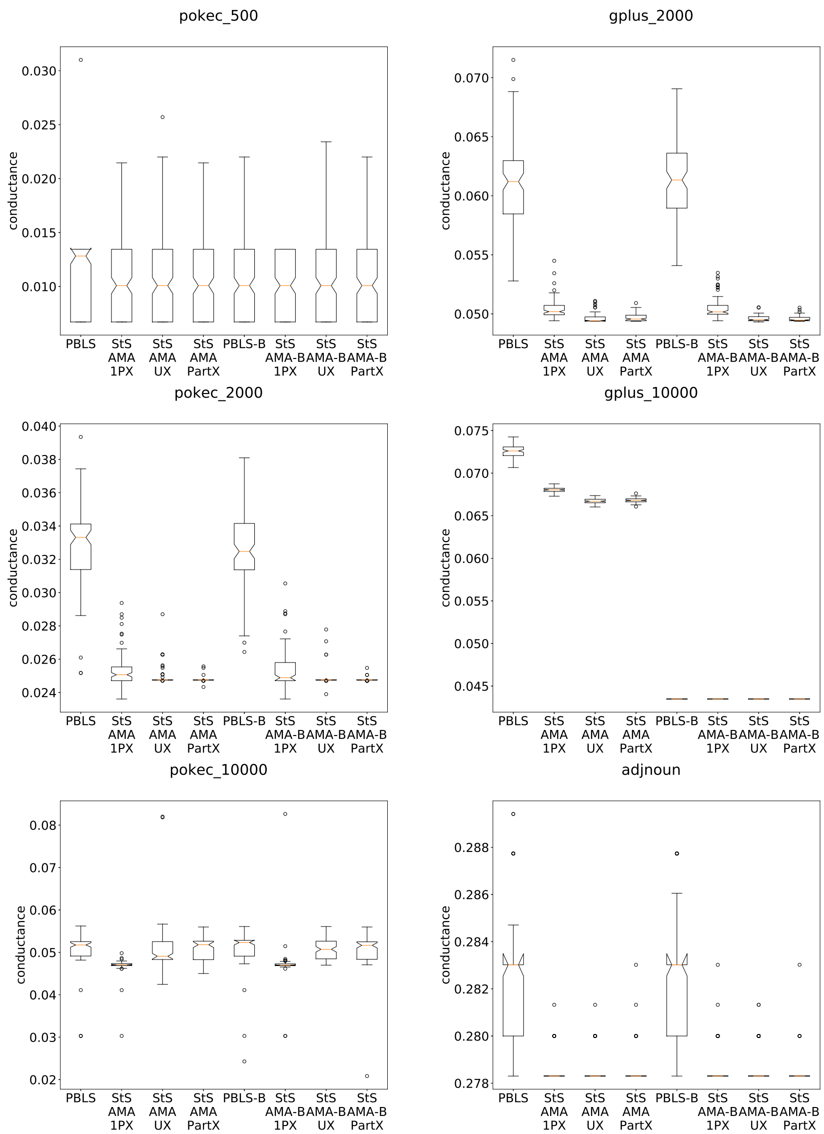

Figure 4: Box-whisker plots depicting the objective values found by all algorithms studied in the short runs with a maximum of 500 generations for selected social network samples and the adjective-noun adjacency network.

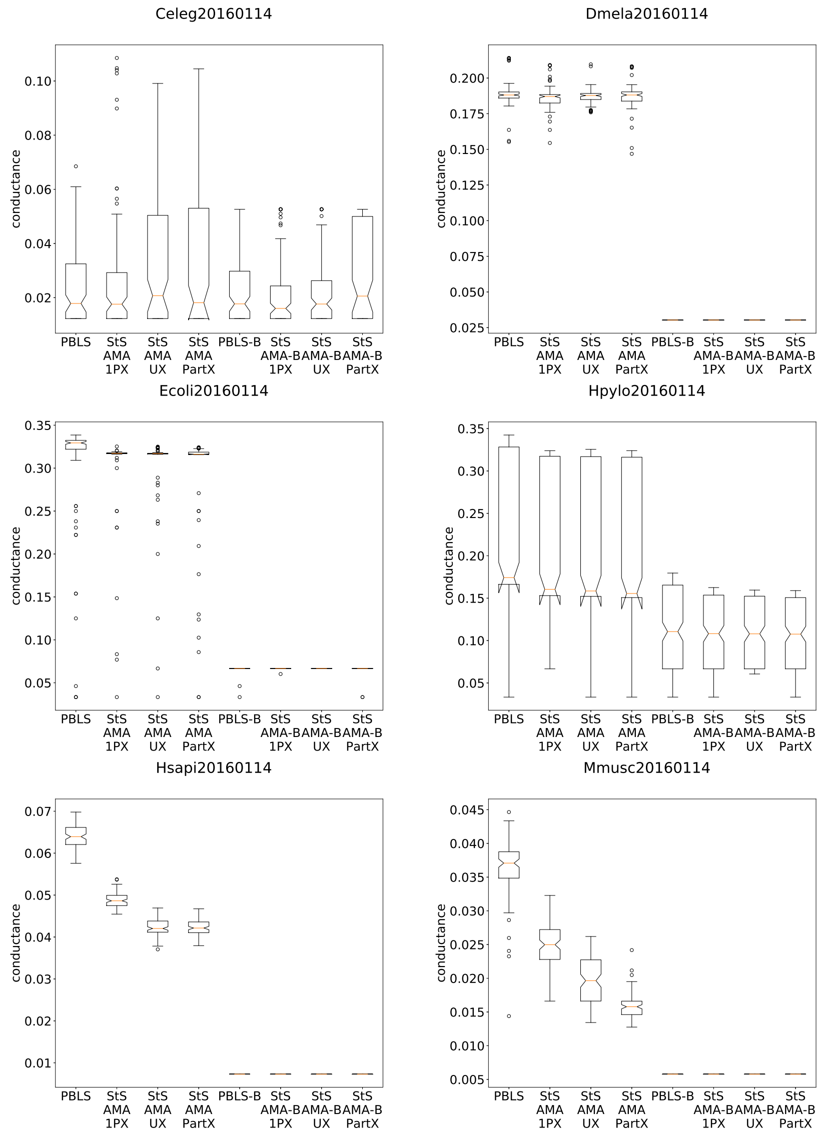

Figure 5: Box-whisker plots depicting the objective values found by all algorithms studied in the short runs with a maximum of 500 generations for selected protein-protein interaction networks.

Table 3 presents the results in short runs obtained for social network samples and an adjective-noun adjacency network [41]. For *gplus_500*, the bridge-based component ensures 100% success rate. For *gplus_2000* and *pokec_2000*, the results are more mixed. StS AMA-B PartX works best on average. However, the results for *pokec_2000* are intriguing in the sense that the variants with one-point crossover were the only ones to produce the best solution in "lucky" runs, even though these were less successful on average. The results for large samples are also quite interesting. For *gplus_10000*, the bridge-based variants performed better again. On the other hand, *pokec_10000* seems to be the instance, for which the algorithms exhibit the clearest probabilistic behaviour. This will be discussed further in the next paragraphs. For *adjnoun*, one can observe that the use of a crossover improves on the success rate.

In Table 4, the results obtained are presented for protein-protein interaction networks from the UCLA database of interacting proteins [37, 38, 39, 40]. For some of the networks, the bridge-based component was particularly helpful. One can observe that the bridge-based variants perform considerably well for *Dmela*20160114, *Hpylo*20160114, *Hsapi*20160114 and *Scere*20160114. For *Celeg*20160114, all algorithms seem to behave probabilistically, producing the best bottleneck roughly in one in three runs. The results for *Ecoli*20160114 were relatively interesting. Even though most algorithms produced the best bottleneck, they did so only occasionally. A similar situation occurs also for *Hpylo*20160114.

For both Table 3 and Table 4, the corresponding box-whisker plots are also presented in Figure 4 and Figure 5, respectively. These also provide some further insights. One can observe that crossover-based algorithms performed much better than the crossover-free PBLS for *gplus_2000* and *pokec_2000*. This indicates that a crossover is beneficial in solving this problem already for moderately large social networks. The distinctive performance of StS AMA 1PX and StS AMA-B 1PX for *pokec_10000* can also be observed. For protein-protein interaction networks, the intriguing nature of *Ecoli*20160114 can be observed on the number of outlier points in its box-whisker plot. Interestingly, some of the plots suggest that the partition crossover was the most reliable (particularly *Hsapi*20160114 and *Mmusc*20160114). However, this seems to depend on a particular instance and contrasts with the success of one-point crossover for *pokec_10000*. One can also note that the bridge-based component notably helped in improving the performance for *Hpylo*20160114. The plot for *Scere*20160114 is omitted here, since it was very flat, only showing the differences between the bridge-based and other algorithms.

Table 5: Experimental results comparing the population-based local search (PBLS) with variants of the steady-state memetic algorithm (StS AMA) with one-point (1PX), uniform (UX) and partition crossovers (PartX) in short runs within a maximum of 500 generations for easy problem instances. Bridge-based variants (-B) of the algorithms are also included in the comparison.

G	algorithm	$\min \Phi(S)$	$E[\Phi(S)]$	success rate
gplus_200	PBLS	**0.02040816**	0.02679298	83 / 100
	StS AMA 1PX	**0.02040816**	0.02547221	85 / 100
	StS AMA UX	**0.02040816**	0.02551323	82 / 100
	StS AMA PartX	**0.02040816**	0.02655511	81 / 100
	PBLS-B	**0.02040816**	**0.02040816**	100 / 100
	StS AMA-B 1PX	**0.02040816**	**0.02040816**	100 / 100
	StS AMA-B UX	**0.02040816**	**0.02040816**	100 / 100
	StS AMA-B PartX	**0.02040816**	**0.02040816**	100 / 100
pokec_500	PBLS	**0.01345291**	0.01353298	99 / 100
	StS AMA 1PX	**0.01345291**	0.01362846	99 / 100
	StS AMA UX	**0.01345291**	0.01353298	99 / 100
	StS AMA PartX	**0.01345291**	0.01377617	96 / 100
	PBLS-B	**0.01345291**	0.01365892	97 / 100
	StS AMA-B 1PX	**0.01345291**	**0.01345291**	100 / 100
	StS AMA-B UX	**0.01345291**	0.01388407	95 / 100
	StS AMA-B PartX	**0.01345291**	0.01373918	96 / 100
*Rnorv*20160114	PBLS	**0.00671141**	0.00693076	96 / 100
	StS AMA 1PX	**0.00671141**	**0.00671141**	100 / 100
	StS AMA UX	**0.00671141**	**0.00671141**	100 / 100
	StS AMA PartX	**0.00671141**	**0.00671141**	100 / 100
	PBLS-B	**0.00671141**	**0.00671141**	100 / 100
	StS AMA-B 1PX	**0.00671141**	**0.00671141**	100 / 100
	StS AMA-B UX	**0.00671141**	**0.00671141**	100 / 100
	StS AMA-B PartX	**0.00671141**	**0.00671141**	100 / 100

Table 6: Experimental results comparing the variants of the steady-state memetic algorithm (StS AMA) with one-point (1PX), uniform (UX) and partition crossovers (PartX) in long runs within a maximum of 10000 generations for the more difficult problem instances. All variants of the algorithms were bridge-based (-B) and used crossover operators.

G	algorithm	$\min \Phi(S)$	$E[\Phi(S)]$	success rate
*gplus*_2000	StS AMA-B 1PX	0.04937560	0.04978619	3 / 30
	StS AMA-B UX	0.04941531	0.04955669	21 / 30
	StS AMA-B PartX	**0.04931558**	**0.04949921**	1 / 30
*pokec*_2000	StS AMA-B 1PX	**0.02360775**	**0.02433515**	13 / 30
	StS AMA-B UX	0.02470694	0.02477831	10 / 30
	StS AMA-B PartX	0.02470694	0.02474227	8 / 30
*gplus*_10000	StS AMA-B 1PX	**0.04347826**	**0.04347826**	30 / 30
	StS AMA-B UX	**0.04347826**	**0.04347826**	30 / 30
	StS AMA-B PartX	**0.04347826**	**0.04347826**	30 / 30
*pokec*_10000	StS AMA-B 1PX	**0.03030303**	**0.04593560**	1 / 30
	StS AMA-B UX	0.04616896	0.04900827	1 / 30
	StS AMA-B PartX	0.04601006	0.04810808	1 / 30
*Celeg*20160114	StS AMA-B 1PX	**0.01226994**	0.01553310	22 / 30
	StS AMA-B UX	**0.01226994**	**0.01489945**	26 / 30
	StS AMA-B PartX	**0.01226994**	0.01624947	19 / 30
*Dmela*20160114	StS AMA-B 1PX	**0.03030303**	**0.03030303**	30 / 30
	StS AMA-B UX	**0.03030303**	**0.03030303**	30 / 30
	StS AMA-B PartX	**0.03030303**	**0.03030303**	30 / 30
*Ecoli*20160114	StS AMA-B 1PX	**0.03333333**	**0.03977778**	1 / 30
	StS AMA-B UX	0.06666667	0.06666667	30 / 30
	StS AMA-B PartX	**0.03333333**	0.06549784	1 / 30
*Hpylo*20160114	StS AMA-B 1PX	**0.14855876**	0.15114257	1 / 30
	StS AMA-B UX	**0.14855876**	0.15137511	1 / 30
	StS AMA-B PartX	**0.14855876**	**0.15045777**	1 / 30
*Hsapi*20160114	StS AMA-B 1PX	**0.00729927**	**0.00729927**	30 / 30
	StS AMA-B UX	**0.00729927**	**0.00729927**	30 / 30
	StS AMA-B PartX	**0.00729927**	**0.00729927**	30 / 30
*Mmusc*20160114	StS AMA-B 1PX	**0.00578035**	**0.00578035**	30 / 30
	StS AMA-B UX	**0.00578035**	**0.00578035**	30 / 30
	StS AMA-B PartX	**0.00578035**	**0.00578035**	30 / 30
*Scere*20160114	StS AMA-B 1PX	**0.07317073**	**0.13362950**	1 / 30
	StS AMA-B UX	0.14285714	0.14285714	30 / 30
	StS AMA-B PartX	0.14285714	0.14285714	30 / 30

Table 5 summarises the results obtained for some of the easier problem instances identified. Remarkably, all eight algorithms studied had the same performance also for a number of instances that seem very easy to solve. For all of these instances, all algorithms achieved solutions of the same quality with 100 / 100 success rate. These results included 0.13108614 for *soc*52, 0.12252964 for *lesmis* [42], 0.10116086 for *football* [4], 0.12820513 for *zachary* [49], 0.165575758 for *celegansneural* [43], 0.06382979 for *dolphins* [44] and 0.04347826 for *polbooks*.

Last but not least, Table 6 summarises the results obtained for the more difficult instances identified with a maximum of 10000 generations. These are of a high interest, to determine whether an extension of the runtime budget can lead to an improvement of the results achieved. One can observe that the results were preserved for *gplus*_10000, *Dmela*20160114, *Hsapi*20160114 and *Mmusc*20160114, mainly due to the bridge-based component. For *gplus*_2000, the dominance of StS AMA-B 1PX has been confirmed with a higher success rate. The best result for *gplus*_2000 was reproduced. However, this was achieved by a different algorithm this time. A similar result was obtained for *Ecoli*20160114, for which the one-point crossover seems to work best. However, one can also observe that the best result obtained for *pokec*_10000 is still worse than that obtained with the shorter limit. This indicates that for some instances, the algorithms may be more suitable as probabilistic sampling routines. In contrast, StS AMA-B 1PX produced a notable improvement in one of the runs, nearly halving the best objective value found in the rest of the experiments.

4.4. Discussion

Figure 6 illustrates the bottlenecks found for the more difficult instances, i.e. those which have not been easily found by all of the algorithms. The bridge-based characteristics are clearly visible for some of the solutions found, including protein-protein interaction networks *Dmela*20160114, *Hsapi*20160114 and *Mmusc*20160114. However, one can also identify several sparse cuts with more than one edge situated between S and $V \setminus S$ that stand out from the drawings, including the solutions found for *Celeg*20160114, *Ecoli*20160114 and *Scere*20160114. Remarkably, the bottlenecks found for *gplus*_2000 and *pokec*_2000 are situated between relatively balanced partitions, the bottlenecks found for *gplus*_10000 and *pokec*_10000 are quite imbalanced, representing a partition into a relatively small community and the

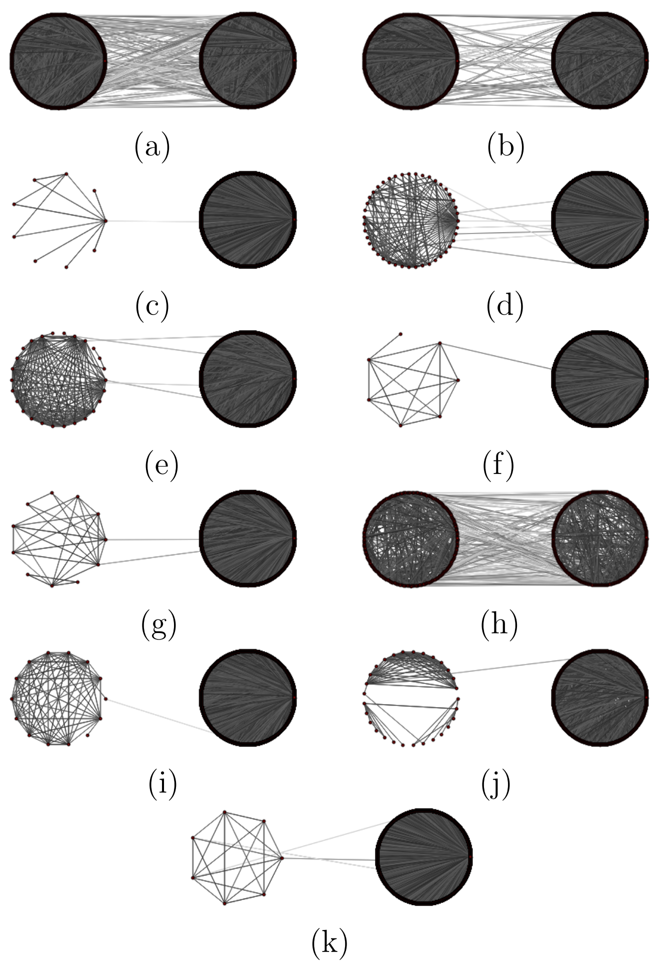

Figure 6: An illustration of the bottlenecks identified for each instance using StS AMA for the more difficult test instances. These includes bottlenecks found for (a) *gplus_200*, (b) *pokec_2000*, (c) *gplus_10000*, (d) *pokec_10000*, (e) *Celeg*20160114, (f) *Dmela*20160114, (g) *Ecoli*20160114, (h) *Hpylo*20160114, (i) *Hsapi*20160114, (j) *Mmusc*20160114 and (k) *Scere*20160114.

Table 7: A comparison of all algorithms tested in producing bottlenecks with best conductance for each instance. Both experiments in short runs and long runs were considered. A success rate information is given as a percentage and represents that the corresponding algorithm was able to find the bottleneck with best conductance found so far in at least one run.

G	PBLS (%)	StS AMA 1PX (%)	StS AMA UX (%)	StS AMA PartX (%)	PBLS-B (%)	StS AMA-B 1PX (%)	StS AMA-B UX (%)	StS AMA-B PartX (%)	StS AMA-B 1PX (long) (%)	StS AMA-B UX (long) (%)	StS AMA-B PartX (long) (%)
$gplus\_2000$							1				3
$pokec\_2000$		8				3			43		
$gplus\_10000$					100	100	100	100	100	100	100
$pokec\_10000$								1			
$Celeg$20160114	31	26	29	34	34	38	37	28	73	87	63
$Dmela$20160114					100	100	100	100	100	100	100
$Ecoli$20160114	3	1	1	2	1			2	3		3
$Hpylo$20160114		1	1					2	3	3	3
$Hsapi$20160114					100	100	100	100	100	100	100
$Mmusc$20160114					100	100	100	100	100	100	100
$Scere$20160114									3		

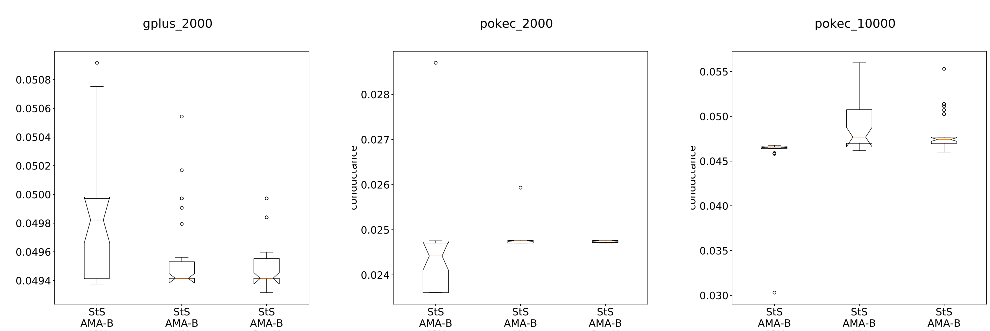

Figure 7: Box-whisker plots depicting the objective values found by StS AMA-B 1PX, StS AMA-B UX and StS AMA-B PartX in the long runs with a maximum of 10000 generations for the social networks. These figures are generated for selected instances, for which the algorithms exhibit notably distinct performance characteristics.

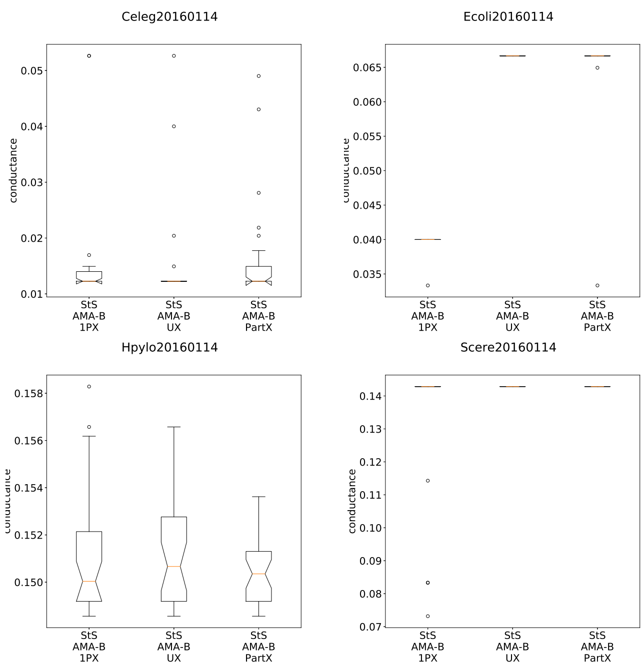

Figure 8: Box-whisker plots depicting the objective values found by StS AMA-B 1PX, StS AMA-B UX and StS AMA-B PartX in the long runs with a maximum of 10000 generations for the protein-protein interaction networks. Similarly to the previous figure, these results were also chosen for their notable characteristics.

rest of the network. These findings support our hypothesis that the high-quality solutions to the problem can have a variety of structural properties and depend on the properties of the entire network.

Regarding the algorithms suitable for finding these partitions, the box-whisker plots presented in Figure 7 and Figure 8 shed light on suitable algorithm choice. These correspond to the computational results presented in Table 6. These results are mixed and there seems to be no golden rule for the choice of the right crossover in particular. While the uniform and partition crossovers work better for *gplus*_2000, the one-point crossover worked for *pokec*_2000. However, the plots obtained for *pokec*_10000, *Ecoli*20160114 and *Scere*20160114 indicate that one-point crossover can sometimes obtain better results that the uniform and partition crossovers were not able to produce. This is exhibited by a presence of several successful runs observed for *pokec*_10000 and *Scere*20160114, as well as the better median performance for *Ecoli*20160114.

To provide a high-level perspective, Table 7 presents a summary of the results of the algorithms in finding the best bottleneck identified in both short and long runs. A success rate value is given as a percentage and denotes that the algorithm was able to produce the best result known for that instance in at least one run. The results confirm that the bridge-based variants are able to solve the problem successfully for a wider variety of instances. PBLS-B, StS AMA-B 1PX and StS AMA-B UX in short runs produced the best results for 6 instances, while StS AMA-B PartX worked for 8 instances. For the long runs, the number of instances with best results remained at 6 for StS AMA-B UX and 8 for StS AMA-B PartX. However, it increased to 9 for StS AMA-B 1PX. This confirms that crossover operators are beneficial in solving this problem. Some of our results suggest that the partition crossover works quite well, even though the simple one-point crossover seems to lead to a good performance in long runs in particular.

5. Conclusions

In this paper a new hybrid bridge-based memetic approach to find bottlenecks in complex networks was proposed. The approach is based on minimising the symmetric variant of the conductance metric. A steady-state adaptive memetic algorithm (StS AMA) is proposed to solve the problem, incorporating a specialised initialisation procedure with a crossover and two local search subroutines. The specialised initialisation procedure includes adaptive generation of potentially imbalanced solutions, as well as the use of bridge identification to decompose the network. All solutions in the population in StS AMA represent local optima at all times in the search process.

To the best of our knowledge, this is the first study aimed at solving the minimum conductance problem in the discrete domain. We focused on the symmetric variant of conductance, for which the optimum is also known in statistical mechanics as the Cheeger constant [20]. While conductance has previously been widely used as metric to evaluate the performance of community detection algorithms [2, 12], it seems that its direct optimisation has so far been studied only using a continuous generalisation of its asymmetric variant [25].

The experimental results obtained indicate that the problem is indeed of a high interest for evolutionary computation techniques. A comparison of StS AMA with crossover operators to a crossover-free variant of the algorithm shows that crossover seems beneficial in solving this problem. Three crossover operators were used, including one-point crossover, uniform crossover and our own partition crossover. Even though all of these operators seem beneficial in general, the magnitude of the impact of a particular crossover seems to depend on the instance structure entirely.

The bottlenecks identified have a variety of structures, including simple bridge-based decompositions, sparse imbalanced cuts into a community and the rest of the network, as well as relatively balanced partitions.

We believe that this study may shed some light on the suitable algorithms and tools to explore the bottlenecks in complex networks. It seems that a proper blend of classical graph algorithms and stochastic optimisation techniques can pave the way towards strong approaches to solve this type of problem. As this seems to be the first study of the minimum conductance problem from the computational discrete optimisation perspective, further studies can be of a high interest, to enrich the variety of literature using conductance as a metric.

Acknowledgement. We acknowledge the Viper High Performance Computing facility of the University of Hull and its support team.

References

[1] U. Brandes, M. Gaertler, D. Wagner, Experiments on graph clustering algorithms, in: European Symposium on Algorithms, Springer, 2003, pp. 568–579.

[2] J. Leskovec, K. J. Lang, M. Mahoney, Empirical comparison of algorithms for network community detection, in: Proceedings of the 19th International Conference on World Wide Web, ACM, 2010, pp. 631–640.

[3] V. D. Blondel, J. L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, Journal of Statistical Mechanics: Theory and Experiment 2008 (10) (2008) P10008.

[4] M. Girvan, M. E. J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences 99 (12) (2002) 7821–7826.

[5] K. A. Hawick, Applying enumerative, spectral and hybrid graph analyses to biological network data, in: International Conference on Computational Intelligence and Bioinformatics (CIB 2011), IASTED, Pittsburgh, USA, 2011, pp. 89–96.

[6] K. A. Hawick, Betweenness centrality metrics for assessing electrical power network robustness against fragmentation and node failure, in: Proceedings of an International Conference on Power and Energy Systems (EuroPES 2012), IASTED, Napoli, Italy, 2012, pp. 186–193.

[7] G. Chen, Z. Y. Dong, D. J. Hill, G. H. Zhang, K. Q. Hua, Attack structural vulnerability of power grids: A hybrid approach based on complex networks, Physica A 389 (2010) 595–603.

[8] K. A. Hawick, Water distribution network robustness and fragmentation using graph metrics, in: Proceedings of an Interntational Conference on Water Resource Management (AfricaWRM 2012), no. 762-037, IASTED, Gabarone, Botswana, 2012, pp. 304–310, CSTN-158.

[9] K. A. Hawick, Betweenness centrality and individual and hub importances in social network graphs, Tech. Rep. CSI-0017, Computer Science, University of Hull, Cottingham Road, Hull HU6 7RX, UK (September 2016).

[10] S. E. Schaeffer, Graph clustering, Computer Science Review 1 (1) (2007) 27–64.

[11] C. Pizzuti, S. E. Rombo, Algorithms and tools for protein–protein interaction networks clustering, with a special focus on population-based stochastic methods, Bioinformatics 30 (10) (2014) 1343–1352.

[12] J. Leskovec, K. J. Lang, A. Dasgupta, M. W. Mahoney, Statistical properties of community structure in large social and information networks, in: Proceedings of the 17th International Conference on World Wide Web, ACM, 2008, pp. 695–704.

[13] J. Leskovec, K. J. Lang, A. Dasgupta, M. W. Mahoney, Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters, Internet Mathematics 6 (1) (2009) 29–123.

[14] C. Pizzuti, Evolutionary computation for community detection in networks: a review, IEEE Transactions on Evolutionary Computation, to appear.

[15] L. Backstrom, D. Huttenlocher, J. Kleinberg, X. Lan, Group formation in large social networks: membership, growth, and evolution, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge discovery and data mining, ACM, 2006, pp. 44–54.

[16] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, P. Spyridonos, Community detection in social media, Data Mining and Knowledge Discovery 24 (3) (2012) 515–554.

[17] A. Metwally, D. Agrawal, A. El Abbadi, Detectives: detecting coalition hit inflation attacks in advertising networks streams, in: Proceedings of the 16th International Conference on World Wide Web, ACM, 2007, pp. 241–250.

[18] G. Bocewicz, Z. Banaszak, I. Nielsen, Reduction of congestion in transport networks with a fractal structure, in: Proceedings of 37th International Conference on Information Systems Architecture and Technology – ISAT 2016 – Part III, Springer International Publishing, Cham, 2017, pp. 189–201.

[19] R. Khandekar, K. Hildrum, S. Parekh, D. Rajan, J. Wolf, K. L. Wu, H. Andrade, B. Gedik, COLA: Optimizing Stream Processing Applications via Graph Partitioning, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 308–327.

[20] E. Arias-Castro, B. Pelletier, P. Pudlo, The normalized graph cut and Cheeger constant: from discrete to continuous, Advances in Applied Probability 44 (4) (2012) 907–937.

[21] J. Šíma, S. E. Schaeffer, On the NP-completeness of some graph cluster measures, in: Proceedings of the 32nd International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM '06, Springer, Berlin / Heidelberg, 2006, pp. 530–537.

[22] S. Chawla, R. Krauthgamer, R. Kumar, Y. Rabani, D. Sivakumar, On the hardness of approximating multicut and sparsest-cut, Computational Complexity 15 (2) (2006) 94–114.

[23] D. Chalupa, A Memetic Algorithm for the Minimum Conductance Graph Partitioning Problem, ArXiv e-prints 1704.02854.

[24] R. E. Tarjan, A note on finding the bridges of a graph, Information Processing Letters 2 (6) (1974) 160–161.

[25] T. Van Laarhoven, E. Marchiori, Local network community detection with continuous optimization of conductance and weighted kernel k-means, Journal of Machine Learning Research 17 (147) (2016) 1–28.

[26] S. Fortunato, Community detection in graphs, Physics reports 486 (3) (2010) 75–174.

[27] D. Chalupa, J. Pospíchal, Metaheuristically optimized multicriteria clustering for medium-scale networks, in: V. Snášel, A. Abraham, E. S. Corchado (Eds.), Soft Computing Models in Industrial and Environmental Applications, SOCO '12, Vol. 188 of Advances in Intelligent Systems and Computing, Springer, Berlin / Heidelberg, 2013, pp. 337–346.

[28] W. Chen, D. Whitley, D. Hains, A. Howe, Second order partial derivatives for NK-landscapes, in: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, GECCO '13, ACM, New York, NY, USA, 2013, pp. 503–510.

[29] R. Tintos, D. Whitley, F. Chicano, Partition crossover for pseudo-boolean optimization, in: Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII, ACM, 2015, pp. 137–149.

[30] F. Chicano, D. Whitley, A. M. Sutton, Efficient identification of improving moves in a ball for pseudo-boolean problems, in: Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12-16, 2014, 2014, pp. 437–444.

[31] D. Whitley, W. Chen, Constant time steepest descent local search with lookahead for NK-landscapes and MAX-kSAT, in: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, ACM, 2012, pp. 1357–1364.

[32] C. Pizzuti, A multiobjective genetic algorithm to find communities in complex networks, IEEE Transactions on Evolutionary Computation 16 (3) (2012) 418–430.

[33] U. Benlic, J. K. Hao, A multilevel memetic approach for improving graph k-partitions, IEEE Transactions on Evolutionary Computation 15 (5) (2011) 624–642.

[34] G. Syswerda, A study of reproduction in generational and steady state genetic algorithms, Foundations of Genetic Algorithms 2 (1991) 94–101.

[35] L. Takac, M. Zabovsky, Data analysis in public social networks, in: International Scientific Conference and International Workshop Present Day Trends of Innovations, 2012, pp. 1–6.

[36] J. Leskovec, A. Krevl, SNAP Datasets: Stanford large network dataset collection.

[37] L. Salwinski, C. S. Miller, A. J. Smith, F. K. Pettit, J. U. Bowie, D. Eisenberg, The database of interacting proteins: 2004 update, Nucleic Acids Research 32 (suppl 1) (2004) D449–D451.

[38] I. Xenarios, E. Fernandez, L. Salwinski, X. J. Duan, M. J. Thompson, E. M. Marcotte, D. Eisenberg, Dip: the database of interacting proteins: 2001 update, Nucleic Acids Research 29 (1) (2001) 239–241.

[39] I. Xenarios, D. W. Rice, L. Salwinski, M. K. Baron, E. M. Marcotte, D. Eisenberg, Dip: the database of interacting proteins, Nucleic Acids Research 28 (1) (2000) 289–291.

[40] I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S. M. Kim, D. Eisenberg, Dip, the database of interacting proteins: a research tool for studying cellular networks of protein interactions, Nucleic Acids Research 30 (1) (2002) 303–305.

[41] M. E. J. Newman, Finding community structure in networks using the eigenvectors of matrices, Physical Review E 74 (036104) (2006) 036104–1–036104–19.

[42] D. E. Knuth, The Stanford GraphBase: A Platform for Combinatorial Computing, Addison-Wesley, Reading, MA, 1993.

[43] D. J. Watts, S. H. Strogatz, Collective dynamics of “small-world” networks, Nature 393 (6684) (1998) 440–442.

[44] D. Lusseau, K. Schneider, O. J. Boisse, P. Haase, E. Slooten, S. M. Dawson, The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations, Behavioral Ecology and Sociobiology 54 (4) (2003) 396–405.

[45] D. Chalupa, P. Balaghan, K. A. Hawick, N. A. Gordon, Computational methods for finding long simple cycles in complex networks, Knowledge-Based Systems 125 (2017) 96–107.

[46] D. Chalupa, C. Blum, Mining k-reachable sets in real-world networks using domination in shortcut graphs, Journal of Computational Science 22 (2017) 1–14.

[47] A. L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.

[48] R. Albert, A.-L. Barabási, Statistical mechanics of complex networks, Reviews of Modern Physics 74 (1) (2002) 47–97.

[49] W. W. Zachary, An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33 (1977) 452–473.