A Fast Exact Algorithm for the Problem of Optimum Cooperation and Structure of Its Solutions

Diana Fanghänel[1] and Frauke Liers[1]

Universität zu Köln, Institut für Informatik, Pohligstraße 1, 50969 Köln, Germany

Abstract. Given a graph $G = (V, E)$ with edge weights $w_e \in \mathbb{R}$, the optimum cooperation problem consists in determining a partition of the graph that maximizes the sum of weights of the edges with nodes in the same class plus the number of the classes of the partition. The problem is also known in the literature as the optimum attack problem in networks. Furthermore, a relevant physics application exists.
In this work, we present a fast exact algorithm for the optimum cooperation problem. Algorithms known in the literature require $|V| - 1$ minimum cut computations in a corresponding network. By theoretical considerations and appropriately designed heuristics, we considerably reduce the numbers of minimum cut computations that are necessary in practice. We show the effectiveness of our method by presenting results on instances coming from the physics application. Furthermore, we analyze the structure of the optimal solutions.

1 Introduction

In this work, we will deal with the following optimization problem. Suppose the benefit of cooperation between two people, say, researchers, is represented by some number. Furthermore, there is a unit gain for each group working on, say, a research project. We search for an optimal cooperation, i.e., want to decide which researchers should collaborate in order to maximize the total benefit, see [1].

In graph-theoretic terms, the problem can be stated as follows. Let a graph $G = (V, E)$ with edge weights $w_e \in \mathbb{R}$ for the edges $e \in E$ be given. Vertices represent the researchers, edge weights correspond to the benefit of cooperation. We want to solve the problem

$$\max\{c_G(A) + w(A) : \ A \subseteq E\}, \tag{1.1}$$

where $w(A) = \sum_{e \in A} w_e$ and $c_G(A)$ is the number of connected components of the induced graph $G(A) = (V, A)$.

The problem was first studied by Cunningham in the context of determining optimum attacks in networks [5]. In this application, the weight w_e can be interpreted as a measure for the effort required by an attacker to destroy edge e. The task is to minimize the difference between the effort of destroying a set of edges and the number of newly generated components of the graph.

Another relevant application is the separation of partition inequalities, as introduced by Baïou, Barahona and Mahjoub [2]. Given a partition $\{S_1, \ldots, S_p\}$ of the node set

V, we denote by $\delta(S_1, \ldots, S_p)$ the set of all edges having endnodes in different sets of the partition. Then, for given real numbers a and b, the inequality $w(\delta(S_1, \ldots, S_p)) \leq ap + b$ is called partition inequality. The latter arise as valid inequalities for a number of combinatorial problems. In order to use them inside a cutting plane algorithm, we need to solve the separation problem that, given edge weigths $w_e \geq 0$, returns a partition violating the inequality, if it exists. Baïou et al. show that the separation problem can be solved by computing an optimal solution of (1.1).

An important model in statistical physics is the Potts model [13]. It has been introduced as a generalization of the so-called Ising model to describe several physical systems. It is a model on a graph where the vertices represent magnetic spins. They are assigned variables that each can take values between $\{1, \ldots, q\}$. Interactions between pairs of spins may be present. The aim is to compute the so-called partition function that encodes the physics of the system. For many relevant physics systems, computing the partition function is a difficult task. However, as pointed out by Juhasz, Rieger, Iglói [15] and Anglès d'Auriac, Iglói, Preissmann and Sebö [1], for big numbers q, determining the dominant contribution in the Potts partition function amounts to solving a problem of type (1.1).

Several solution algorithms for determining optimum cooperations or optimum attacks have been presented in the literature. As it is not hard to see that the function to be maximized is supermodular, any algorithm for submodular function minimization could be used to solve the problem. By now several polynomial algorithms [4, 7] are known to solve this task. However, the specific properties of the problem allow the usage of algorithms with better worst-case asymptotic running time.

Cunningham [5] developed the first combinatorial algorithm for the optimum attack problem that is based on $|E|$ minimum cut computations in an associated network. Thereafter, the worst-case running time of the algorithm was decreased to $|V| - 1$ minimum cut computations by Baïou, Barahona and Mahjoub, and Barahona [2, 3]. Anglès d'Auriac et al. [1] built upon the existing work and presented an algorithm that also needs $|V| - 1$ minimum cut computations but is easier to implement. They presented some experimental results for instances coming from the physics application.

In this article we present an algorithm that is based on the one of Anglès d'Auriac et al. but has a better average running time, as by graph-theoretic considerations only a fraction of the minimum cut computations are necessary in practice. The paper is self-contained. In Section 2 we define the necessary concepts. We also provide optimality conditions and theoretical results that prove the correctness of our algorithm. In Section 3 we explain the algorithm of Anglès d'Auriac et al. Thereafter, we propose a modification yielding better performance. In Section 4 we explore the influence of changes in the edge weights to the set of optimal solutions. Furthermore, we discuss geometric properties of the so-called regions of stability. In Section 5 we explain details of the implementation and present experimental results on instances coming from the physics applications. Furthermore, we compare our algorithm with the original method of Anglès d'Auriac et al. It turns out that for many instances the running times can be reduced considerably.

2 Definitions and Theoretical Concepts

2.1 Introduction

Let a graph $G = (V, E)$ with edge weights $w_e \in \mathbb{R}$ for all edges $e \in E(G)$ be given. Then

$$\max \left\{ f_{G,w}(A) = c_G(A) + \sum_{e \in A} w_e : \ A \subseteq E(G) \right\} \tag{2.2}$$

is called Potts problem and $f_{G,w}$ Potts function. $c_G(A)$ is the number of all connected components of the graph $G(A) = (V, A)$. $w(A)$ denotes the sum of weights of the edges in A. Let $\Psi^*_G(w)$ denote the set of all optimal solutions of the Potts problem, i.e.,

$$\Psi^*_G(w) := \operatorname{Argmax} \left\{ f_{G,w}(A) = c_G(A) + \sum_{e \in A} w_e : \ A \subseteq E(G) \right\}. \tag{2.3}$$

Solving the optimum cooperation problem means to determine one of the solutions from $\Psi^*_G(w)$. For a preprocessing step, we observe the following. Obviously, deleting edges from G with nonpositive weights yields an equivalent problem. Analogously, for an edge $e \in E(G)$ with $w_e \geq 1$, there always exists an optimal solution containing e. Thus, we obtain an equivalent problem by contracting all edges with weight at least 1. W.l.o.g., we assume $w_e \in (0, 1)$ for all $e \in E(G)$.

2.2 Potts Partitions

In this subsection we introduce the concept of Potts partitions which will be helpful for further considerations.

Definition 2.1. *Let $\mathcal{P} = \{P_1, \ldots, P_k\}$ be a partition of the nodes $V(G)$ of $G = (V, E)$. Then $\mathcal{P}$ is called a Potts partition if the sets P_i induce connected subgraphs of G for all indices $i = 1, \ldots, k$. With $\mathfrak{P}_G$ we denote the set of all Potts partitions of G.*

To each Potts partition $\mathcal{P} = \{X_1, \ldots, X_k\}$ we assign a value $F_{G,w}(\mathcal{P}) := |\mathcal{P}| + w(\mathcal{P})$, where $|\mathcal{P}| := k$ denotes the number of classes of $\mathcal{P}$ and $w(\mathcal{P}) := \sum_{i=1}^{k} \{w_e : e \in E(G)$ and the endpoints of e are in the same class$\}$. Thus, if $E(\mathcal{P})$ is the induced edge set $E(\mathcal{P}) := \{e \in E(G) :$ the endpoints of e are in the same class of $\mathcal{P}\}$, we obtain equality of the function values $f_{G,w}(E(\mathcal{P})) = F_{G,w}(\mathcal{P})$ for all Potts partitions $\mathcal{P} \in \mathfrak{P}_G$.

Definition 2.2. *A Potts partition is called optimal if it is an optimal solution of the problem*

$$\max_{\mathcal{P} \in \mathfrak{P}_G} F_{G,w}(\mathcal{P}).$$

The reason why Potts partitions are relevant for solving the Potts problem (2.2) is the following. Let A^* be an optimal solution of the latter. It is easy to see that each connected component of $G(A^*)$ contains all edges of G it induces. In fact, let $X_1, \ldots, X_k \subseteq V(G)$ be the vertices of the connected components of the induced graph $G(A^*) = (V, A^*)$. If there exists an edge $e = (u, v) \in E(G) \backslash A^*$ with both u, v contained in the same set X_i, adding e to A^* increases the value of the Potts function, in contradiction to the optimality of A^*. Therefore, it is possible to consider node partitions instead of edge sets. We will see in the following that the notion of Potts partitions eases the exposition of certain statements. In the following, we will either speak of edge sets or of the corresponding Potts partition, whatever is more natural in the context.

2.3 The Supermodularity of the Potts Function

It is not hard to see that the Potts function is supermodular [1]. We define the concept of supermodularity next.

Definition 2.3. *Let a function $f : 2^E \to \mathbb{R}$ be given. If the inequality*

$$f(A_1 \cup A_2) + f(A_1 \cap A_2) \geq f(A_1) + f(A_2) \tag{2.4}$$

holds for all subsets $A_1, A_2 \subseteq E$, the function f is called supermodular. If $(-f)$ is supermodular, f is called submodular.

Lemma 2.1 ([1]). *The Potts function $f_{G,w}$ is supermodular.*

Submodular function minimization occurs in a huge number of different applications, see, e.g., [4]. Whereas it was already known for some time that the problem itself can be solved in polynomial time [10, 11], only recently, strongly polynomial combinatorial algorithms could be designed [4, 7, 14, 19]. For solving (2.2), we could therefore use any algorithm for minimizing submodular functions. However, exploiting the structure of the Potts function yields a faster solution algorithm, for example the optimum cooperation algorithm. Furthermore, under certain conditions the problem can be decomposed into smaller problems of the same type. Moreover, certain subgraphs can be shrunk into a single node. Applying the two principles of decomposition and shrinking yields a solution algorithm which is on average considerably faster than the basic algorithm for the optimum cooperation problem. In the next section, we introduce the basic concepts.

2.4 Decomposing the Problem

Because of the supermodularity of the Potts function, there exists an optimum solution on G containing edge sets which are optimum for its induced subgraphs. This is stated in the next lemma which is a generalization from a lemma given in [1].

In the following, $G(U)$ with $U \subseteq V(G)$ denotes the subgraph of G that is induced by the vertex set $U \neq \emptyset$.

Lemma 2.2. *Let a graph $G = (V, E)$ and a vertex set $U \subset V(G)$ with $\emptyset \neq U \neq V(G)$ be given. Suppose that $A_1 \subseteq E(G_1)$ resp. $A_2 \subseteq E(G_2)$ are optimal solutions of (2.2) for the induced subgraphs $G_1 = G(U)$ resp. $G_2 = G(V\backslash U)$.*

Then there exists an optimal solution $A^* \subseteq E(G)$ of the Potts problem on the original graph G with $A^* \supseteq A_1 \cup A_2$.

Proof. Supermodularity implies

$$f_{G,w}(A \cup A_1) \geq f_{G,w}(A) + f_{G,w}(A_1) - f_{G,w}(A \cap A_1)$$

for all edge sets $A \subseteq E(G)$. Furthermore, since A_1 and $A_1 \cap A$ are subsets of $E(G_1)$ it holds

$$f_{G,w}(A_1) = f_{G_1,w}(A_1) + |V\backslash U| \geq f_{G_1,w}(A_1 \cap A) + |V\backslash U| = f_{G,w}(A \cap A_1).$$

Then these inequalities imply $f_{G,w}(A \cup A_1) \geq f_{G,w}(A)$ for all edges $A \subseteq E(G)$. Thus, there exists some set $A_1^* \in \Psi_G^*(w)$ with $A_1^* \supseteq A_1$.

Similarly, there exists some set $A_2^* \in \Psi_G^*(w)$ with $A_2^* \supseteq A_2$.

Using the equality $f_{G,w}(A_1^*) = f_{G,w}(A_2^*)$ and the supermodularity of $f_{G,w}$ it is easy to prove that $A^* = A_1^* \cup A_2^*$ is an optimal solution for G. □

Thus, we can obtain an optimal solution for G by a divide-and-conquer approach in which we first solve the problem for the smaller graphs $G(U)$ and $G(V\backslash U)$ and add further edges to the union of the optima for the smaller graphs. A special case is the choice of node sets having cardinality 1 for which Lemma 2.2 was proven in [1].

In order to formulate Lemma 2.2 using Potts partitions, let $X_1, \ldots, X_k$ be the vertex sets of the connected components of $G(A_1)$ and $Y_1, \ldots, Y_l$ the vertex sets of the connected components of $G(A_2)$. Then $\mathcal{P} = \{X_1, \ldots, X_k, Y_1, \ldots, Y_l\}$ is a Potts partition of $V(G)$. Lemma 2.2 says that in an optimal solution A^* for G the vertex sets of the components of $G(A^*)$ are either classes or the union of classes of $\mathcal{P}$.

This fact will be important for the development of a fast solution algorithm. Furthermore, we need some an inversion of Lemma 2.2, given in the following.

Lemma 2.3. *Let G be a graph with an optimal Potts partition $\mathcal{P} = \{X_1, \ldots, X_k\}$. Further, let $I \subset \{1, \ldots, k\}$ be a nonempty index set inducing a subgraph $G' = G(\bigcup_{i \in I} X_i)$ of G. Then $\mathcal{P}' = \{X_i : i \in I\}$ is an optimal Potts partition of G'.*

Proof. Assume the lemma is not valid. Then there exists a Potts partition $\bar{\mathcal{P}}$ for the graph G' with $F_{G',w}(\bar{\mathcal{P}}) > F_{G',w}(\mathcal{P}')$. Consequently,

$$\begin{aligned} F_{G,w}(\mathcal{P}) &= |\mathcal{P}| + \sum_{i=1}^{k} \sum_{x,y \in X_i} w(x,y) = (k - |I|) + \sum_{i \notin I} \sum_{x,y \in X_i} w(x,y) + F_{G',w}(\mathcal{P}') \\ &< (k - |I|) + \sum_{i \notin I} \sum_{x,y \in X_i} w(x,y) + F_{G',w}(\bar{\mathcal{P}}) = F_{G,w}(\mathcal{P}'') \end{aligned} \tag{2.5}$$

for a partition $\mathcal{P}'' = \{X_i : i \notin I\} \cup \bar{\mathcal{P}}$ for G. This is a contradiction to the optimality of $\mathcal{P}$. □

Knowing the decomposition lemma (2.2), we are interested in possible choices of the sets U. It turns out that sets defining cuts of small weight are important.

Theorem 2.1. *Let $U \subseteq V(G)$ with cut $\delta(U) := \{(v_1, v_2) \in E(G) : \ v_1 \in U, \ v_2 \notin U\}$ be given. Assume that the weight $w(\delta(U))$ of the cut is $w(\delta(U)) \leq 1$. Then there exists an optimal solution A^* of the Potts problem (2.2) with $\delta(U) \cap A^* = \emptyset$.*

Proof. Let be given some optimal solution $A \subseteq E(G)$ with $A \cap \delta(U) \neq \emptyset$. Then it holds $c_G(A \backslash \delta(U)) \geq c_G(A) + 1$. Furthermore, $w(A \cap \delta(U)) \leq w(\delta(U)) \leq 1$. Hence, it holds

$$\begin{aligned} f_{G,w}(A \backslash \delta(U)) &= c_G(A \backslash \delta(U)) + w(A) - w(A \cap \delta(U)) \\ &\geq c_G(A) + w(A) + 1 - w(A \cap \delta(U)) \\ &\geq f_{G,w}(A), \end{aligned} \tag{2.6}$$

i.e., $A^* = A \backslash \delta(U)$ is also an optimal solution for which $\delta(U) \cap A^* = \emptyset$. □

For the case $U = \{v\}$ with $v \in V(G)$ we obtain the following trivial corollary.

Corollary 2.1. *Suppose it holds $w(\delta(v)) \leq 1$ for some vertex $v \in V(G)$. Then there exists an optimal solution A^* of the Potts problem (2.2) with $A^* \cap \delta(v) = \emptyset$.*

Therefore, whenever we find a partition of the nodes of G into sets U and $V \backslash U$ with cut weight at most 1, we can decompose the problem into two problems of the same type, one defined on $G(U)$ and the other on $G(V \backslash U)$, that can be solved independently. An optimal solution for G then consists of the union of the edge sets that are optimum for $G(U)$ and $G(V \backslash U)$. We state this formally in the following corollary.

Corollary 2.2. *Let be given some set $U \subseteq V(G)$ with $w(\delta(U)) \leq 1$. Further let A_1 resp. A_2 be optimal solutions for the graphs $G_1 = G(U)$ and $G_2 = G(V \backslash U)$. Then $A_1 \cup A_2$ is an optimal solution for G.*

Proof. Following Theorem 2.1, there exists an optimal solution A^* for G with $A^* \cap \delta(U) = \emptyset$. Then it holds

$$\begin{aligned} f_{G,w}(A^*) &= f_{G_1,w}(A^* \cap E(G_1)) + f_{G_2,w}(A^* \cap E(G_2)) \\ &\leq f_{G_1,w}(A_1) + f_{G_2,w}(A_2) = f_{G,w}(A_1 \cup A_2), \end{aligned} \tag{2.7}$$

i.e., $A_1 \cup A_2$ is an optimal solution for G. □

2.5 Contracting Subgraphs

Apart from decomposing the problem, it is also possible to contract certain subgraphs of G. As a preprocessing step, we already contract edges $e \in E(G)$ with weight at least 1 before starting a solution algorithm. We now generalize this procedure to the contraction of subgraphs of G.

In our context, contracting a subgraph $G(U)$ means replacing U by a supernode v_U. For all nodes $v \in V \backslash U$, we replace the set of edges (v, u) with $u \in U$ by a single edge

$e = (v, v_U)$ with weight $w_e := \sum_{(v,u):u \in U} w_{(v,u)}$. Edges (u_1, u_2) with $u_1, u_2 \in U$ are deleted.

Contracting an edge set $E' \subseteq E(G)$ means contracting the set of all vertices incident to these edges to a supernode, dealing with loops and multiple edges as before. G/U resp. G/E' denote the graphs generated by contracting the vertex set $U \subseteq V(G)$ resp. the edge set $E' \subseteq E(G)$ in G. If in a node set P a subset $U \subseteq P$ is replaced by a supernode, we write P/U. In the next theorem we state the condition under which a subgraph can be contracted.

Theorem 2.2. *Let a Potts partition $\mathcal{P} = \{P_1, \ldots, P_k\}$ of $V(G)$ be given. Furthermore, let $U \subseteq P_k$ be chosen such that $\tilde{\mathcal{P}} = \{U\}$ is optimal for $G(U)$. Then $\mathcal{P}$ is an optimal Potts partition for G if and only if $\mathcal{P}' = \{P_1, \ldots, P_{k-1}, P_k/U\}$ is optimal for G/U.*

Proof. Let $\mathcal{P}$ be an optimal partition of G and let $\mathcal{P}^*$ be optimal for the graph G/U. Then it holds $F_{G/U,w}(\mathcal{P}^*) \geq F_{G/U,w}(\mathcal{P}')$. Decontracting the set U in $\mathcal{P}^*$ we obtain a Potts partition $\mathcal{P}^{**}$ for G with

$$F_{G,w}(\mathcal{P}^{**}) = F_{G/U,w}(\mathcal{P}^*) + w(U) \geq F_{G/U,w}(\mathcal{P}') + w(U) = F_{G,w}(\mathcal{P}). \tag{2.8}$$

Consequently, the optimality of $\mathcal{P}$ implies $F_{G/U,w}(\mathcal{P}^*) = F_{G/U,w}(\mathcal{P}')$, i.e., $\mathcal{P}'$ is an optimal partition of G/U.

Now we assume that $\mathcal{P}'$ an optimal Potts partition G/U. Because of Lemma 2.2 and the optimality of the Potts partition $\tilde{\mathcal{P}}$ for $G(U)$ there exists an optimal partition $\mathcal{P}^{**} = \{P_1^{**}, \ldots, P_l^{**}\}$ of G with $U \subseteq P_l^{**}$. Hence, $\mathcal{P}^* = \{P_1^{**}, \ldots, P_{l-1}^{**}, P_l^{**}/U\}$ is a Potts partition of G/U. Using the optimality of $\mathcal{P}'$ we now obtain

$$F_{G,w}(\mathcal{P}^{**}) = F_{G/U,w}(\mathcal{P}^*) + w(U) \leq F_{G/U,w}(\mathcal{P}') + w(U) = F_{G,w}(\mathcal{P}), \tag{2.9}$$

i.e., $\mathcal{P}$ is an optimal Potts partition of G. □

Theorem 2.2 can be used for the computation of an optimal solution as follows: If we find a subset $U \subseteq V(G)$ such that $\{U\}$ is optimum on $G(U)$, we proceed by solving the problem on G/U instead of G. G/U has a smaller number of vertices than G. Furthermore, some edge weights are increased by the contraction. As soon as the weight of an edge exceeds the value 1, it can be contracted.

2.6 Optimality Conditions

In this subsection we investigate optimality conditions for the Potts problem. We closely follow the considerations in [1, 13] and generalize some results from [1].

Let $U \subseteq V(G)$ be a vertex set such that $\mathcal{P}' = \{U\}$ is an optimal Potts partition for $G(U)$, and let $\mathcal{P}'' = \{X_1, \ldots, X_k\}$ be optimal for $G(V \backslash U)$. Then $\mathcal{P} = \{X_1, \ldots, X_k, U\}$ is a Potts partition for G. It turns out that we can obtain an optimal partition for G from the latter by merging U with classes of $\mathcal{P}$. In the next lemma we state how such a union of classes changes the value of a Potts partition.

Lemma 2.4 ([1]). *Let $\mathcal{P}$ and $\mathcal{P}^*$ be two Potts partitions of G with $\mathcal{P}^* = (\mathcal{P} \backslash \mathcal{W}) \cup \{\bigcup_{X_i \in \mathcal{W}} X_i\}$ for a set $\mathcal{W} \subseteq \mathcal{P}$. Then $f_{G,w}(\mathcal{P}^*) = f_{G,w}(\mathcal{P}) - |\mathcal{W}| + 1 + w(E(\mathcal{W}))$, where $E(\mathcal{W})$ is the set of all edges between the vertex sets $X_i, X_j \in \mathcal{W}$, $i \neq j$.*

For completeness, we include the proof of this lemma.

Proof. If in the Potts partition $\mathcal{P}$ all sets $X_i \in \mathcal{W}$ are replaced by the union of these sets, the new partition contains $|\mathcal{W}| - 1$ classes less. However, on the other side more edges are induced. These are exactly the edges of $E(\mathcal{W})$ with the weight $w(E(\mathcal{W}))$. Hence the lemma is valid. □

In Lemma 2.4 the partition $\mathcal{P}^*$ is supposed to be a Potts partition. Thus, the graph $G(\bigcup_{X_i \in \mathcal{W}} X_i)$ has to be connected. Otherwise, the lemma cannot be applied. Using Lemma 2.4 we obtain the following trivial corollary.

Corollary 2.3. *Let $\mathcal{P}$ be a Potts partition of G. Then $\mathcal{P}$ is optimal if and only if the inequality $|\mathcal{W}| - 1 - w(E(\mathcal{W})) \geq 0$ is valid for all $\mathcal{W} \subseteq \mathcal{P}$ for which $G(\bigcup_{X_i \in \mathcal{W}} X_i)$ is connected.*

Now we can prove the following theorem.

Theorem 2.3. *Let $U \subseteq V(G)$ be chosen such that $\{U\}$ is optimal for $G(U)$. Furthermore, let $\mathcal{P}'$ be an optimal Potts partition for $G' = G(V \backslash U)$ with induced edge set $A^* \subseteq E(G')$. Then, for $\mathcal{W} \subseteq \mathcal{P}'$ that attains the minimal value of $|\mathcal{W}| - w(E(\mathcal{W} \cup \{U\}))$, the edge set $A^* \cup E(\mathcal{W} \cup \{U\})$ is an optimal solution of the Potts problem for G.*

Proof. Let $U \subseteq V(G)$ with $\{U\}$ being an optimal solution of $G(U)$ and let $\mathcal{P}' = \{X_1, \ldots, X_k\}$ be optimal for $G' = G(V \backslash U)$.

Then, because of Lemma 2.4 and Corollary 2.3 there exists a set $\mathcal{W}^* \subseteq \mathcal{P}$ with $\mathcal{P} = \{X_1, \ldots, X_k, U\}$ such that $(\mathcal{P} \backslash \mathcal{W}^*) \cup \{\bigcup_{X \in \mathcal{W}^*} X\}$ is an optimal Potts partition of G. The latter induces an edge set $A^* \cup E(\mathcal{W}^*)$. We only have to prove that, w.l.o.g., $\{U\} \in \mathcal{W}^*$ can be assumed.

Suppose $\{U\} \notin \mathcal{W}^*$. Then it holds $|\mathcal{W}^*| - 1 - w(E(\mathcal{W}^*)) \geq 0$ because of Corollary 2.3 and the optimality of $\mathcal{P}'$ for $G(V \backslash U)$. Using the optimality of $(\mathcal{P} \backslash \mathcal{W}^*) \cup \{\bigcup_{X_i \in \mathcal{W}^*} X_i\}$ for G and Lemma 2.4 we obtain $|\mathcal{W}^*| - 1 - w(E(\mathcal{W}^*)) = 0$. But then we can also choose $\mathcal{W}^* = \{U\}$ to obtain an optimal Potts partition. □

The optimality conditions in this subsections have been proven in [1] restricted to the special case $U = \{v\}$ for some vertex $v \in V(G)$.

3 The Algorithm

In this section we present algorithms for solving the Potts problem (2.2). As all of them are based on the algorithm of Preissmann, Sebö et al. [1] we first introduce their original method. In the subsections thereafter we propose modifications yielding considerable decrease in running time on practical instances.

3.1 An Exact Algorithm

The optimal cooperation algorithm proposed in [1] uses the optimality conditions given in Lemma 2.3, applied to node sets of cardinality 1. It starts with a trivial solution for a subgraph $G(U)$ with $|U| = 1$. Iteratively, nodes are added to U one after one. In each step of the algorithm an optimum solution on $G(U)$ is known. After having added a new node to U, an optimum solution on the new induced graph is computed from the former optimum by calculating a minimum $s - t$ cut in an associated network. If $U = V$, the solution at hand is optimum for the whole graph G.

Let us assume that problem (2.2) has been solved on $G(U)$ with $U \subset V(G)$, yielding $\mathcal{P}_U = \{X_1, \ldots, X_k\}$ as optimal partition. Then we contract the sets $X_1, \ldots, X_k$ as described in Subsection 2.5 and work with the possibly smaller graph $\mathrm{shr}(G(U))$. Hence we assume, w.l.o.g., that the sets $X_1, \ldots, X_k$ are singletons and $G(U) = \mathrm{shr}(G(U))$.

Let $v \in V(G)\backslash U$. For maintaining optimality, we have to compute a set $W \subseteq U \cup \{v\}$, $v \in W$, that minimizes the function

$$b(W) := |W| - 1 - w(E(W)). \tag{3.10}$$

Thus we have to solve the problem

$$\min\{b(W) : \ W \subseteq U \cup \{v\}, \ v \in W\}. \tag{3.11}$$

In [1] it was shown that such a set W can be determined by computing a minimum cut in an associated directed network $D_{U,v}$. For completeness, we briefly summarize the procedure.

We define a directed graph $D_{U,v}$ with weighted edges as follows:

1. Let $V(D_{U,v}) := U \cup \{v\} \cup \{s, t\}$ with new vertices $s \neq t$.
2. Let $E(D_{U,v})$ be the set of all directed edges (u, u') and (u', u) with $u, u' \in U \cup \{v\}$ and $(u, u') \in E(G)$. To each edge $(u, u') \in E(D_{U,v})$ a weight $c_{(u,u')} := \frac{1}{2} w_{(u,u')}$ is assigned.
3. To each vertex $u \in U \cup \{v\}$ the real number

$$p(u) := \frac{1}{2} \sum_{u' \in U \cup \{v\}, (u,u') \in E(G)} w_{(u,u')}$$

is assigned.

   (a) If $p(u) > 1$, we add a directed edge (s, u) to the set $E(D_{U,v})$ with weight $c_{(s,u)} = p(u) - 1$.

   (b) If $p(u) < 1$, we add a directed edge (u, t) to the set $E(D_{U,v})$ with weight $c_{(u,t)} = 1 - p(u)$.

Obviously, for each subset $W \subseteq U \cup \{v\}$ the set $W \cup \{s\}$ corresponds to an $s - t$ cut $\delta(\{s\} \cup W)$ in $D_{U,v}$.

Lemma 3.1 ([1]). *Let $W \subseteq U \cup \{v\}$. Then it holds*

$$c(\delta(\{s\} \cup W)) = |W| - w(E(W)) + \kappa = b(W) + \kappa + 1 \tag{3.12}$$

with a constant $\kappa = c(\delta(s)) = \sum_{(s,u) \in E(D_{U,v})} c_{(s,u)}$.

Lemma 3.1 states that we have to compute a minimum $s-t$ cut $W \cup \{s\}$ with $v \in W$. Then the set W has a minimal value $b(W)$. To ensure $v \in W$ we can shrink the vertex set $\{s, v\}$ in $D_{U,v}$ or add an edge (s, v) with infinite weight to $D_{U,v}$.

Using the results from Lemma 3.1 we explain the optimal cooperation algorithm which was proposed in [1].

Algorithm 3.1. *(optimal cooperation algorithm [1].)*

Input: *A graph $G = (V, E)$ with edge weights $w_e \in (0, 1)$ for all edges $e \in E(G)$.*
Output: *An optimal Potts partition $\mathcal{P}$ of G.*

1. *Set $U := \emptyset$ and $\mathcal{P} := \emptyset$.*
2. *Choose a vertex $v \in V \backslash U$.*
3. *Determine the directed graph $D_{U,v}$ and add an edge (s, v) with infinite weight to $D_{U,v}$.*
4. *Determine the set $W \subseteq U \cup \{v\}$ which solves problem (3.11) by computing a minimum $s-t$ cut in $D_{U,v}$.*
5. *Set $U := U \cup \{v\}$ and construct the new optimal Potts partition $\mathcal{P}$.*
6. *Shrink the vertex set W in G and set $U := U/W$.*
7. *If $U \neq V(G)$, go to step 2.; else STOP.*

In the following example we explain the flow of the algorithm in detail.

Example 3.1. Let be given the graph of Figure 1 with edge weights $w_e = 0.8$ for all edges $e \in E(G)$.

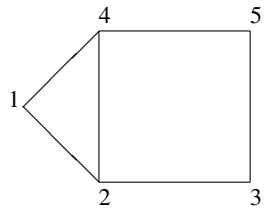

Fig. 1. Graph in which an optimum Potts partition has to be computed.

Let $U = \{1, 2, 3\}$. Then it is easy to see that $\mathcal{P} = \{\{1\}, \{2\}, \{3\}\}$ is an optimal Potts partition of $G(U)$, i.e., the empty set is an optimal solution of (2.2). Choosing vertex $v = 4$, we obtain the network as shown in Figure 2.

$W = \{1, 2, 4\}$ is a minimum s-t cut in $D_{U,v}$. The new optimal Potts partition $\mathcal{P} = \{\{1, 2, 4\}, \{3\}\}$ is computed by unifying with $\{v\}$ all classes in $\mathcal{P}$ in which the vertices $u \in W \backslash \{v\}$ are contained. Now we contract W to a supernode w which yields a new set $U = \{w, 3\}$ and a new graph G/W as can be seen in Figure 3.

with all edge weights equal to 0.8. In the next iteration we choose vertex $v = 5$ and again construct the network $D_{U,v}$. Computing a minimum $s-t$ cut we obtain the set $W = \{w, 3, 5\}$. We construct a new partition by unifying the class $\{1, 2, 4\}$ that is

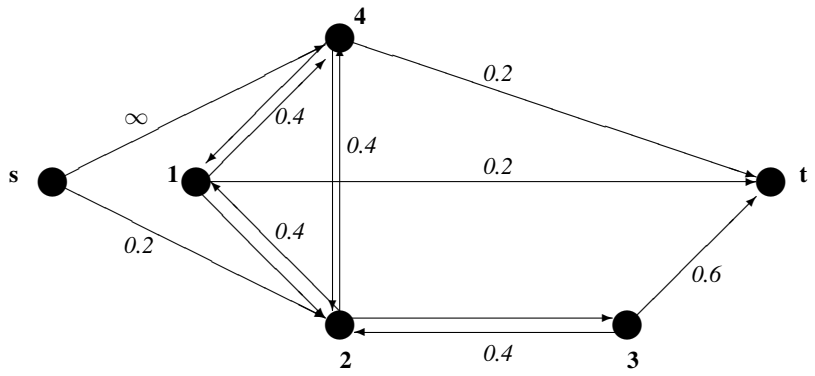

Fig. 2. Choosing $v = 4$, we get the following directed network in which a minimum s-t cut has to be computed.

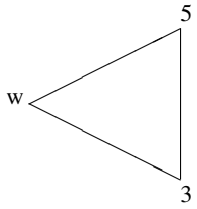

Fig. 3. Graph G/W after shrinking the node $W = \{1, 2, 4\}$ determining a minimum s-t cut in $D_{U,v}$.

represented by the vertex w, the class $\{3\}$ and $\{v\}$. The algorithm stops with the optimal Potts partition $\mathcal{P} = \{1, 2, 3, 4, 5\}$ of G. Consequently, $E(G)$ is an optimal solution of (2.2) for G.

Example 3.1 shows that there is a one-to-one correspondence between the classes of the Potts partition $\mathcal{P}$ and the vertices of $G(U)$, the sets W being contracted. It is important to know for each vertex $u \in U$ the accompanying class of the partition $\mathcal{P}$. Contractions are not performed in the partition, but only in the graph and the set U.

In Algorithm 3.1 $|V(G)| - 1$ minimum cut problems are solved in graphs with $|U| + 3$ vertices and at most $2(|E(G(U \cup \{v\})| + |U| + 3)$ many arcs. If, e.g., the Goldberg-Tarjan algorithm [8] is used for the computation of the minimum $s-t$ cuts, the algorithm has a worst case running time of $O(|V(D_{U,v})|^2\sqrt{|E(D_{U,v})|})$ in each of the $|V(G)| - 1$ iterations. Thus, Algorithm 3.1 has polynomial running time. However, its performance depends strongly on the size of the directed graphs $D_{U,v}$ and the number of minimum cut computations.

In the following subsection we present ideas for improving the running time of the algorithm.

3.2 Enhancement of the Basic Exact Algorithm

In order to reduce the running time in practice, we explore several ideas. We first briefly summarize the necessary conditions and present their justifications subsequently.

Firstly, if we know that for a given vertex $u \in V(G)$ there exists an optimal solution of the Potts problem that does not contain any edge incident to u, we do not need to choose u in Step 2. of the algorithm. According to our experience, this often reduces the number of minimum cut computations and the cardinality of the set U. Moreover, this also implies a reduction of the size of the networks $D_{U,v}$ and thus better running times of the minimum cut algorithms. Secondly, assume a vertex $u \in V(G)$ has been choosen in Step 2. of the algorithm. If we know that $W = \{u\}$ is an optimal solution of problem (3.11) we can skip Steps 3. and 4. of the algorithm. Finally, assume we know a set $U \subseteq V(G)$ such that $\{U\}$ is an optimal Potts partition of $G(U)$. Then we can contract the set U in G and apply the algorithm to the smaller graph G/U (cf. Theorem 2.2).

The first idea is strongly related to Corollary 2.1. Before we start Algorithm 3.1 we determine for each vertex $v \in V(G)$ the value $w(\delta(v))$. All vertices $v \in V(G)$ with $w(\delta(v)) \leq 1$ are saved in a set $notU$. Thereafter, Algorithm 3.1 is applied to $G\backslash notU$. Let $\mathcal{P} = \{X_1, \ldots, X_k\}$ be an optimal Potts partition of $G\backslash notU$ and let $notU = \{v_1, \ldots, v_l\}$. Then

$$\mathcal{P}^* = \{X_1, \ldots, X_k, \{v_1\}, \ldots, \{v_l\}\} \tag{3.13}$$

is an optimal Potts partition of G, which is a direct consequence of Corollary 2.1.

The second improvement of Algorithm 3.1 works analogously. Let $v \in V(G)\backslash U$ for a given subset $U \subset V(G)$. With

$$\delta_U(v) := \{e \in E(G) : \ e = (v, u), \ u \in U\} \tag{3.14}$$

we denote the set of all edges, where one of the endnodes is v and the other one is contained in U. Furthermore, let A^* be an optimal solution of the Potts problem for

$G(U)$ and let $w(\delta_U(v)) \leq 1$. Then A^* is also an optimal solution for $G(U \cup \{v\})$. This is equivalent to the following lemma.

Lemma 3.2. *Let $v \in V(G)\backslash U$ for a given nonempty subset $U \subset V(G)$. Further, let $w(\delta_U(v)) \leq 1$. Then $W = \{v\}$ is an optimal solution of problem (3.11).*

Thus, if the conditions of the Lemma 3.2 are satisfied, we do not need to determine the set W by computing a minimum $s-t$ cut in the network $D_{U,v}$.

In the third approach we ask for a set $U \subseteq V(G)$ such that $\{U\}$ is an optimal Potts partition of $G(U)$. To find such a set in an efficient way, we need knowledge about the solution of the problem for special classes of graphs. It turns out that possible candidates are for instance cycles and complete graphs.

Suppose that we know a graph G' such that $\{V(G')\}$ is an optimal Potts partition of G'. Then we need to develop a fast algorithm that finds a subgraph of G that is equivalent to G'.

Assuming we already know how this can be done we give an outline of the improved algorithm. In order to be able to reconstruct the optimum partition from the shrunk graph, we assign a set $S(v)$ to the vertices $v \in V$ that saves the original nodes shrunk into supernode v. These sets are updated throughout the algorithm.

Algorithm 3.2. *(improved algorithm)*

Input: *A graph $G = (V, E)$ with edge weights $w_e \in (0, 1)$ for all edges $e \in E(G)$.*

1. *Compute the set $notU$ and delete all vertices $v \in notU$ from G.*
2. *Set $U := \emptyset$ and $\mathcal{P} := \emptyset$.*
3. *Choose $v \in V\backslash U$ and set $S(v) := \{v\}$.*
4. *If $w(\{(u, v) \in E(G) : \ u \in U\}) \leq 1$, set $U := U \cup \{v\}$ and $\mathcal{P} := \mathcal{P} \cup \{S(v)\}$ and proceed with Step 3.*
5. *While a subset $W \subseteq U \cup \{v\}$, $|W| > 1$, is found with $v \in W$ such that $\{W\}$ is an optimal partition for $G(W)$, do the following:*
   Set $U := U\backslash W$ and $\mathcal{P} := \mathcal{P}\backslash \bigcup_{u \in (W\backslash\{v\})} S(u)$. Contract W in G to a supernode v_W and set $S(v_W) := \bigcup_{u \in W} S(u)$. Identify $v := v_W$.
6. *Construct $D_{U,v}$ and determine $W \subseteq U \cup \{v\}$ that solves (3.11) by computing a minimum $s-t$ cut in $D_{U,v}$.*
   Then, update U, $\mathcal{P}$, G, and v, analogously to Step 5.
7. *While there exists an edge $e = (u, v) \in E(G)$ with $w_e \geq 1$ do:*
   If $u \notin U$, shrink e in G and set $S(v) := S(v) \cup \{u\}$. If $u \in U$, set $W := \{u, v\}$ and update U, $\mathcal{P}$, G, and v, analogously to Step 5.
8. *If in Step 7. an edge $e = (u, v)$ was found with $u \notin U$, proceed with Step 4; else set $U := U \cup \{v\}$, $\mathcal{P} := \mathcal{P} \cup \{S(v)\}$, and proceed with Step 9.*
9. *If $U \neq V(G)$, go to Step 3.*
10. **Output:** *The optimum partition $\mathcal{P} := \mathcal{P} \cup \{\{v\} : \ v \in notU\}$.*

Why is Step 8. of the improved algorithm important? Assume W has been determined by computing a minimum $s-t$ cut in the network $D_{U,v}$. Then there might exist edges $e = (v, u)$ in G/W with $w_e = 1$ and $u \in U$. In this case, the partition we obtain after shrinking these edges is still optimal for $G(U \cup \{v\})$. However, there possibly

exist edges $e = (v, u)$ in G/W with $w_e \geq 1$ and $u \notin U$. Then, after contracting these edges, in general $\mathcal{P} \cup \{S(v)\}$ is not an optimal partition for $G(U \cup \{v\})$. Therefore, we have to repeat the iteration by using vertex v.

In Step 5. of Algorithm 3.2 we iteratively search for a subset $W \subseteq U \cup \{v\}$ with $|W| > 1$ and $v \in W$ such that $\{W\}$ is optimal for $G(W)$. This also includes the search for edges having weight larger than or equal to one. In case the search is successful, contracting the corresponding set decreases the size of the networks and the number of minimum cut computations in the subsequent iterations.

In the following subsection we present theoretical results together with search algorithms providing such kind of subsets.

3.3 Exact Solutions on Specific Subgraphs

In this subsection we provide explicitely the solution of the Potts problem for some special graph structures. Theorem 2.2 says that if we find such a structure as a subgraph $G' = (V', E')$ in G, we can use the optimal solution of the problem for G' to find an optimal solution for the original graph G. Furthermore, we can shrink in G the subgraph G' to a single node and continue working on the smaller graph G/V'. We start with a result on forests.

Lemma 3.3. *Let $G = (V, E)$ be a forest with edge weights $w_e \in (0; 1)$ for all edges $e \in E(G)$. Then $A^* = \emptyset$ is the unique optimal solution of the Potts problem (2.2).*

Proof. For each nonempty set $A \subseteq E(G)$ it holds

$$f_{G,w}(A) = c_G(A) + \sum_{e \in A} \underbrace{w_e}_{<1} < c_G(A) + |A| = |V(G)| = f_{G,w}(\emptyset), \tag{3.15}$$

since for each subset $A \subseteq E(G)$ the graph $G(A) = (V, A)$ is a forest with $|V(G)|$ vertices. Consequently, $A^* = \emptyset$ is an optimal solution of problem (2.2) for G. □

For application in Step 5 of the improved algorithm, the above result is a negative one, as no candidate forest will have an optimum solution that contains all edges. However, forests are subgraphs of many graphs, and the above result yields a helper theorem in the investigation of more difficult structures, for example cycles. Next, we show that under certain conditions an optimum solution on a cycle contains all edges.

Theorem 3.1. *1. Let a cycle C^n with $n \geq 3$ vertices and edge weights $w_e \in (0, 1)$ for all $e \in E(C^n)$ be given. If $\sum_{e \in E(C^n)} w_e \geq n - 1$, the edge set $E(C^n)$ is an optimal solution of (2.2).*

2. *Let F_2 denote a cycle C^4 consisting of four edges with a chord e_0 inducing two triangles with edge sets $\{e_0, e_1, e_2\}$ and $\{e_0, e_3, e_4\}$, resp. Let $w_e \in (0, 1)$ for all $e \in E(F_2)$. Then, $\{V(F_2)\}$ is an optimal partition for F_2 if the weights satisfy $w_{e_0} + w_{e_1} + w_{e_2} + w_{e_3} + w_{e_4} \geq 3$, $w_{e_1} + w_{e_2} \geq 1$ and $w_{e_3} + w_{e_4} \geq 1$.*

Proof. We will prove Theorem 3.1 in detail only for cycles. The proof for the graph F_2 is straight forward by comparing all possible edge sets.

Let $A^* \subseteq E(C^n)$ be an optimal solution of problem (2.2). Assume it is $0 < |A^*| < n$. Then Lemma 2.3 implies that A^* is also an optimal solution for the subgraph $G' = G(A^*)$. $G' = G(A^*)$ is a forest, as $|A^*| < n$. $0 < |A^*|$ is a contradiction to the previous result on forests.

Thus it holds $A^* = \emptyset$ or $A^* = E(C^n)$. The latter implies the lemma, as the inequality $\sum_{e \in E(C^n)} w_e \geq n - 1$ is equivalent to

$$f_{C^n,w}(\emptyset) = n \leq \sum_{e \in E(C^n)} w(e) + 1 = f_{C^n,w}(E(C^n)). \tag{3.16}$$

□

In the following example we demonstrate how Theorem 3.1 can be applied for the solution of the Potts problem.

Example 3.2. Let be given the graph G from Figure 4 with vertex set

$$V(G) = \{x_1, \ldots, x_4, y_1, \ldots, y_4, z_1, \ldots, z_4\}.$$

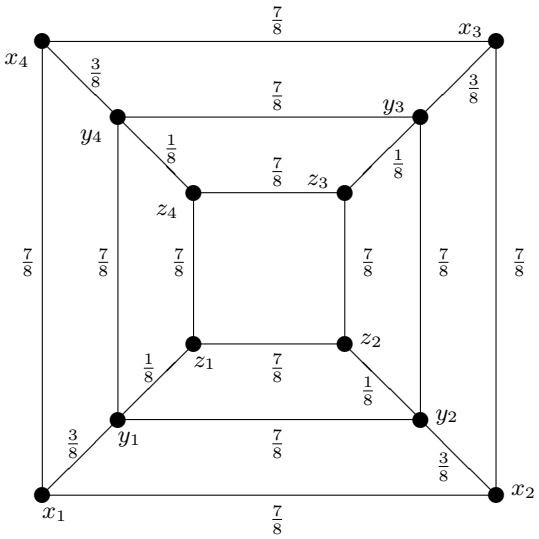

Fig. 4. Graph G/W after shrinking the node $W = \{1, 2, 4\}$ determining a minimum s-t cut in $D_{U,v}$.

The edge weights can be read off from Figure 4.

Let us consider the sets $X = \{x_1, x_2, x_3, x_4\}$, $Y = \{y_1, y_2, y_3, y_4\}$ and $Z = \{z_1, z_2, z_3, z_4\}$. G contains the cycles $G_1 = G(X)$, $G_2 = G(Y)$ and $G_3 = G(Z)$ as induced subgraphs. Theorem 3.1 implies that the edge sets $E(G_1)$, $E(G_2)$ resp. $E(G_3)$

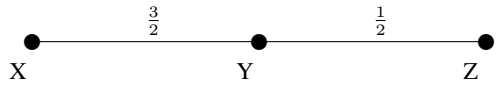

Fig. 5. Graph $\mathrm{shr}(G)$ after shrinking the sets X, Y, Z.

are optimal for G_1, G_2 resp. G_3. We contract the sets X, Y and Z. Then the problem is reduced to the investigation of the graph $\mathrm{shr}(G)$ displayed in Figure 5.

Then, obviously $\bar{A} = \{(X, Y)\}$ is an optimal solution for $\mathrm{shr}(G)$. Using Theorem 2.2 we obtain the following optimal solution for G

$$A^* = E(G)\backslash\{(y_i, z_i) : \ i = 1, \ldots, 4\}.$$

Instead of subsequent minimum cut computations on associated networks, we could solve the toy problem by subsequent application of the theorems outlined above.

The cycles we found in Example 3.2 have been induced subgraphs in G. In general, the subgraphs we find in a graph are often not induced ones. In the following lemma we show that this fact does not cause problems.

Lemma 3.4. *Let $U \subseteq V(G)$ be given and $\mathcal{P}^* = \{U\}$ an optimal Potts partition for $G(U)$ with edges weights w'_e for all $e \in E(G)$. Then $\mathcal{P}^* = \{U\}$ is also optimal for $G(U)$ with edges weights w_e chosen as $w_e \geq w'_e$ for all $e \in E(G)$.*

Proof. Assume that there exists an optimal Potts partition $\mathcal{P} = \{P_1, \ldots, P_k\}$ of $G' = G(U)$ with $F_{G',w}(\mathcal{P}) > F_{G',w}(\mathcal{P}^*)$. Then obviously it holds $E(\mathcal{P}) \subseteq E(\mathcal{P}^*)$. Let $E' = E(\mathcal{P}^*)\backslash E(\mathcal{P})$. Then $F_{G',w}(\mathcal{P}) > F_{G',w}(\mathcal{P}^*)$ implies

$$w(\mathcal{P}^*) + 1 < w(\mathcal{P}) + k \tag{3.17}$$

$$w'(E') \leq w(E') = w(\mathcal{P}^*) - w(\mathcal{P}) < k - 1 \tag{3.18}$$

$$F_{G',w'}(\mathcal{P}^*) = w'(\mathcal{P}^*) + 1 < w'(\mathcal{P}) + k = F_{G',w'}(\mathcal{P}). \tag{3.19}$$

This is a contradiction to the optimality of $\mathcal{P}^*$ for the edge weights w'_e. □

Assume G contains a subgraph G' with $V(G) = V(G')$ and $\{V(G')\}$ being optimal for G'. Then the consideration of G' equals that of G, where the weights are chosen as $w'_e := w_e$ for all $e \in E(G')$ and $w'_e := 0$ for all edges $e \in E(G)\backslash E(G')$. Thus Lemma 3.4 states that $\{V(G)\}$ is an optimal partition of G.

Given these statements, a subgraph satisfying one of the above-mentioned conditions can be contracted.

In the following we present algorithms that, given a graph G with edge weights $w_e \in (0; 1)$ for all $e \in E(G)$, determine subgraphs C^k and F_2 in G that satisfy the conditions of Theorem 3.1. We start the considerations with the search for cycles $C = (V_C, E_C)$ satisfying the condition $\sum_{e \in E_C} w_e \geq |V_C| - 1$. Let G^- denote a copy of G,

where edge weigths are chosen as $w_e^- := 1 - w_e$ for all $e \in E(G)$. Then the condition to be satisfied can be expressed as

$$\sum_{e \in E(G)} w_e^- \leq 1. \tag{3.20}$$

Cycles satisfying the latter can be computed by shortest-path computations in G^- as follows.

For each edge $e = (v, u) \in E(G)$, we temporarily delete e from G^- and search for a shortest path $P = (V_P, E_P)$ in G^- from v to u. We use the algorithm of Dijkstra [6]. If the weight of the path does not exceed $1 - w_e^-$, P together with e forms a cycle satisfying the condition. This method is an exact cycle search, i.e., if there exists a cycle satisfying the condition, the algorithm finds it.

In practice many of the cycles satisfying (3.20) are cycles C^3 with only three nodes which can be found by enumeration. Similarly, candidate subgraphs F_2 are determined by enumeration.

In the next section we discuss the structure of the solution sets. Subsequently, we provide implementation details and present experimental results.

4 Properties of Solution Sets and Regions of Stability

In the Potts problem, in general, the optimal solutions are not unique. Thus, for $w \geq 0$ it is interesting to investigate the structure of the solution sets. In Lemma 3.4 we have proved that if $\{V\}$ is an optimal partition, it stays optimal if we increase the edge weights of the graph. This result will be generalized in this section. We also explore the geometric properties of the so-called regions of stability, that are defined as the set of all edge weights for which a given Potts partition is optimal.

4.1 The Structure of the Optimal Sets

In this subsection we investigate the solution sets

$$\Psi_G^*(w) := \{A \subseteq E(G) : \ f_{G,w}(A) \geq f_{G,w}(B) \text{ for all } B \subseteq E(G)\} \tag{4.21}$$

for the Potts problem (2.2). As for $w > 0$ there is a one-to-one correspondence between the optimal solutions and the optimal Potts partitions (see Subsection 2.2), we also consider the set

$$\Psi_{\mathfrak{P}_G}^*(w) := \{\mathcal{P}^* \in \mathfrak{P}_G : \ F_{G,w}(\mathcal{P}^*) \geq F_{G,w}(\mathcal{P}) \text{ for all } \mathcal{P} \in \mathfrak{P}_G\}. \tag{4.22}$$

These solution sets are point-to-set mappings with respect to the vector of the edge weights $w \in \mathbb{R}_+^{|E(G)|}$. Thus, we are interested in the structure of $\Psi_G^*(w)$ and $\Psi_{\mathfrak{P}_G}^*(w)$ for all possible choices of w and in the way the solution sets change when modifying the edge weights.

First we mention the following well-known result which is also true for general supermodular functions.

Lemma 4.1. *Let $A, B \in \Psi_G^*(w)$ be optimal solutions for $w \in \mathbb{R}_+^{|E(G)|}$. Then $A \cap B$ and $A \cup B$ are also optimal solutions, i.e., $A \cup B, A \cap B \in \Psi_G^*(w)$.*

The above lemma is a special case of the following more general result.

Theorem 4.1. *Let be given some vectors of edge weights $w^1, w^2 \in \mathbb{R}_+^{|E(G)|}$ with $w_e^1 \geq w_e^2$ for all edges $e \in E(G)$. Further, let $A_1 \in \Psi_G^*(w^1)$ and $A_2 \in \Psi_G^*(w^2)$ be some optimal solutions for the edge weights w^1 resp. w^2. Then it is*

$$A_1 \cup A_2 \in \Psi_G^*(w^1) \quad \textit{and} \quad A_1 \cap A_2 \in \Psi_G^*(w^2).$$

Proof. As A_1 resp. A_2 are optimal, it holds

$$c_G(A_1) + w^1(A_1) \geq c_G(A_1 \cup A_2) + w^1(A_1 \cup A_2) \quad \text{and} \tag{4.23}$$
$$c_G(A_2) + w^2(A_2) \geq c_G(A_1 \cap A_2) + w^2(A_1 \cap A_2). \tag{4.24}$$

Adding these inequalities yields

$$c_G(A_1) + c_G(A_2) + w^1(A_1) + w^2(A_2) \geq c_G(A_1 \cup A_2) + c_G(A_1 \cap A_2) \\ + w^1(A_1 \cup A_2) + w^2(A_1 \cap A_2). \tag{4.25}$$

Additionally it is $c_G(A_1) + c_G(A_2) \leq c_G(A_1 \cap A_2) + c_G(A_1 \cup A_2)$ since the function $c_G(\cdot)$ is supermodular [1]. Together with inequality (4.25) it follows

$$\begin{aligned} w^1(A_1) + w^2(A_2) &\geq w^1(A_1 \cup A_2) + w^2(A_1 \cap A_2), \\ w^2(A_2) - w^2(A_1 \cap A_2) &\geq w^1(A_1 \cup A_2) - w^1(A_1), \\ w^2(A_2 \backslash A_1) &\geq w^1(A_2 \backslash A_1). \end{aligned}$$

Because of $w^1 \geq w^2$ this implies equality in all inequalities above, especially in (4.23) and (4.24). Hence it is $A_1 \cup A_2 \in \Psi_G^*(w^1)$ and $A_1 \cap A_2 \in \Psi_G^*(w^2)$. □

Assume we have computed an optimal solution A_2 for edge weights w^2. Then, if we increase the edge weights, there always exists an optimal solution that contains A_2 as a subset. Let $\mathcal{P}_2 = \{Y_1, \ldots, Y_l\}$ be the Potts partition corresponding to A_2. Then in terms of Potts partitions this implies that, when increasing the edge weights, there always exists an optimal partition where its classes are the unions of the Y_j. Therefore, we can shrink the classes of $\mathcal{P}_2$ in G and repeat the computations with the larger weights for the potentially smaller shrunk graph.

Reversely, assume we have computed an optimal solution A_1 for edge weights w^1. Then, there always exists a subset of A_1 that is optimal for the decreased edge weights. Let $\mathcal{P}_1 = \{X_1, \ldots, X_k\}$ be the Potts partition corresponding to A_1. Then this implies that we obtain an optimal solution by solving the Potts problem separately with the smaller edge weights for the graphs $G(X_1), \ldots, G(X_k)$ and unifying their optimal solutions.

For some special cases we can simplify the computation of the new optimal solution further. This is stated in the following easy corollary of Theorem 4.1.

Corollary 4.1. *Let $w^1, w^2 \in \mathbb{R}_+^{|E(G)|}$ with $w^1 \geq w^2$. Further, let $A_1 \in \Psi_G^*(w^1)$ and $A_2 \in \Psi_G^*(w^2)$.*

1. *If* $w_e^1 = w_e^2$ *for all edges* $e \in A_1$*, it is* $A_1 \in \Psi_G^*(w^2)$.
2. *If* $w_e^1 = w_e^2$ *for all edges* $e \notin A_2$*, it is* $A_2 \in \Psi_G^*(w^1)$.

To rewrite Theorem 4.1 in terms of Potts partitions we first need to discuss the lattice properties [9, 20] of the set $\mathfrak{P}_G$.

Definition 4.1. *Let* $\mathcal{P}_1 = \{X_1, \ldots, X_k\}$ *and* $\mathcal{P}_2 = \{Y_1, \ldots, Y_l\}$ *be some Potts partitions. Then we say* $\mathcal{P}_1 \preceq \mathcal{P}_2$ *iff for all indices* $i = 1, \ldots, k$ *there exists some* j *with* $X_i \subseteq Y_j$.

Obviously it holds $\mathcal{P}_1 \preceq \mathcal{P}_2$ iff it is $E(\mathcal{P}_1) \subseteq E(\mathcal{P}_2)$ for the induced edge sets.

$(\mathfrak{P}_G, \preceq)$ is a partial ordering on the set of all Potts partitions. We say that two Potts partitions $\mathcal{P}_1, \mathcal{P}_2 \in \mathfrak{P}_G$ are comparable if $\mathcal{P}_1 \preceq \mathcal{P}_2$ or $\mathcal{P}_2 \preceq \mathcal{P}_1$. Otherwise, $\mathcal{P}_1$ and $\mathcal{P}_2$ are called incomparable which is denoted by $\mathcal{P}_1 || \mathcal{P}_2$.

To each pair of Potts partitions $\mathcal{P}_1, \mathcal{P}_2$ we can assign an infimum $\mathcal{P}_1 \wedge \mathcal{P}_2$ and a supremum $\mathcal{P}_1 \vee \mathcal{P}_2$, i.e., $(\mathfrak{P}_G, \preceq)$ is a lattice. At this, the infimum $\mathcal{P}_1 \wedge \mathcal{P}_2$ is the Potts partition that is induced by the edge set $E(\mathcal{P}_1) \cap E(\mathcal{P}_2)$. The supremum $\mathcal{P}_1 \vee \mathcal{P}_2$ is induced by $E(\mathcal{P}_1) \cup E(\mathcal{P}_2)$.

This implies the following corollary of Theorem 4.1.

Corollary 4.2. *Let* $w^1, w^2 \in \mathbb{R}_+^{|E(G)|}$ *with* $w^1 \geq w^2$*. Further, let* $\mathcal{P}_1 \in \Psi_{\mathfrak{P}_G}^*(w^1)$ *and* $\mathcal{P}_2 \in \Psi_{\mathfrak{P}_G}^*(w^2)$*. Then we have*

$$\mathcal{P}_1 \vee \mathcal{P}_2 \in \Psi_{\mathfrak{P}_G}^*(w^1) \quad \textit{and} \quad \mathcal{P}_1 \wedge \mathcal{P}_2 \in \Psi_{\mathfrak{P}_G}^*(w^2).$$

The results of Corollary 4.2 can also be viewed in the more general context of general parametric supermodular problems on lattices. Amongst others, this setting is studied in detail in the monograph [20] by Topkis. In fact, Theorem 2.8.2 from [20] can be specified appropriately so that Corollary 4.2 results.

Assume two Potts partitions $\mathcal{P}_1 = \{X_1, \ldots, X_k\}$ and $\mathcal{P}_2 = \{Y_1, \ldots, Y_l\}$ are given. Then the supremum $\mathcal{P}_1 \vee \mathcal{P}_2$ is computed from $\mathcal{P}_1$ by substituting iteratively two classes X_{i_1}, X_{i_2} by their union if there exists some j with $X_{i_1} \cap Y_j \neq \emptyset$ and $X_{i_2} \cap Y_j \neq \emptyset$.

To construct the infimum $\mathcal{P}_1 \wedge \mathcal{P}_2$ we consider the partition $\mathcal{P}_\cap = \{X_i \cap Y_j : X_i \cap Y_j \neq \emptyset\}$. This partition has an induced edge set $E(\mathcal{P}_\cap) = E(\mathcal{P}_1) \cap E(\mathcal{P}_2)$ but is not necessarily a Potts partition. We obtain the infimum $\mathcal{P}_1 \wedge \mathcal{P}_2$ by substituting the classes of $\mathcal{P}_\cap$ by the components of the subgraphs $G(X_i \cap Y_j)$.

4.2 Regions of Stability

In this subsection we investigate the geometric structure of the so-called regions of stability. We consider some arbitrary Potts partition $\mathcal{P}^*$ and ask for the set of all edge weights $w \in \mathbb{R}_+^{|E(G)|}$ such that $\mathcal{P}^*$ is optimal.

Definition 4.2. *Let be given some* $\mathcal{P}^* \in \mathfrak{P}_G$*. Then the set*

$$R(\mathcal{P}^*) = \{w \in \mathbb{R}_+^{|E(G)|} : \mathcal{P}^* \in \Psi_{\mathfrak{P}_G}^*(w)\}$$

is called region of stability for $\mathcal{P}^*$.

This is an inverse concept to the solution sets since for all edge weights $w \in \mathbb{R}_+^{|E(G)|}$ it holds

$$\Psi^*_{\mathfrak{P}_G}(w) = \{\mathcal{P} \in \mathfrak{P}_G : \ w \in R(\mathcal{P})\}. \tag{4.26}$$

First we state an alternative formula for the regions of stability in the next easy lemma.

Lemma 4.2. *Let $\mathcal{P}^* \in \mathfrak{P}_G$. Then,*

$$R(\mathcal{P}^*) = \big\{w \in \mathbb{R}_+^{|E(G)|} : \sum_{e\in E(\mathcal{P})} w_e - \sum_{e\in E(\mathcal{P}^*)} w_e \le |\mathcal{P}^*| - |\mathcal{P}| \quad \forall \mathcal{P} \in \mathfrak{P}_G\big\}. \tag{4.27}$$

Proof. Let $w \in \mathbb{R}_+^{|E(G)|}$. Then it is $w \in R(\mathcal{P}^*)$ if and only if $\mathcal{P}^* \in \Psi^*_{\mathfrak{P}_G}(w)$. This is equivalent to the following inequalities.

$$\begin{aligned} F_{G,w}(\mathcal{P}^*) &\ge F_{G,w}(\mathcal{P}) && \forall \mathcal{P} \in \mathfrak{P}_G \\ \sum_{e\in E(\mathcal{P})} w_e - \sum_{e\in E(\mathcal{P}^*)} w_e &\le |\mathcal{P}^*| - |\mathcal{P}| && \forall \mathcal{P} \in \mathfrak{P}_G. \end{aligned}$$

□

Lemma 4.2 states that each region of stability is given by a finite (but exponentially large) number of linear inequalities. Thus the stability regions are convex polyhedrons and so convex and closed. Now we prove that they are nonempty and therefore full-dimensional.

Theorem 4.2. *For all Potts partitions $\mathcal{P} \in \mathfrak{P}_G$ the regions of stability have a nonempty interior, i.e.,* $\operatorname{int} R(\mathcal{P}) \neq \emptyset$.

Proof. Let $\mathcal{P}^* \in \mathfrak{P}_G$ and $w \in \mathbb{R}_+^{|E(G)|}$ be defined as

$$w_e = \begin{cases} 2 & \text{if } e \in E(\mathcal{P}^*) \\ 0 & \text{otherwise.} \end{cases}$$

Assume there exists some $\mathcal{P} \in \Psi^*_{\mathfrak{P}_G}(w)$ with $F_{G,w}(\mathcal{P}) \ge F_{G,w}(\mathcal{P}^*)$. Let $P^* = \{X_1, \dots, X_k\}$ and $P = \{Y_1, \dots, Y_l\}$. Because of the specific structure of w we have

$$w(E(\mathcal{P})) = \sum_{i=1}^{k}\sum_{j=1}^{l} w(X_i \cap Y_j) = w(E(\mathcal{P} \wedge \mathcal{P}^*)),$$

where $w(X_i \cap Y_j)$ denotes the weight of all edges with both endnodes in $X_i \cap Y_j$. Then $\mathcal{P} \in \Psi^*_{\mathfrak{P}_G}(w)$ implies $c_G(\mathcal{P}) = c_G(\mathcal{P} \wedge \mathcal{P}^*)$, and thus, $\mathcal{P} = \mathcal{P} \wedge \mathcal{P}^*$. Therefore, $\mathcal{P} \preceq \mathcal{P}^*$. Let

$$I_i = \{j \in \{1, \dots, l\} : \ Y_j \subseteq X_i\}, \quad i = 1, \dots, k.$$

Then it is $\bigcup_{j\in I_i} Y_j = X_i$ for all $i = 1, \dots, k$. Since the graphs $G(X_i)$ are connected, it holds

$$|E(X_i)| \ge \sum_{j\in I_i} |E(Y_j)| + (|I_i| - 1), \qquad i = 1, \dots, k.$$

If there exists an index i with $|I_i| > 1$, we obtain

$$\begin{aligned} |I_i| + \sum_{j \in I_i} w(Y_j) &= |I_i| + 2 \sum_{j \in I_i} |E(Y_j)| \\ &\leq 2|E(X_i)| - |I_i| + 2 \\ &< 2|E(X_i)| + 1 \\ &= w(X_i) + 1. \end{aligned}$$

This is a contradiction to $\mathcal{P} \in \Psi^*_{\mathfrak{P}_G}(w)$, since $\mathcal{P}' = (\mathcal{P} \backslash \{Y_j : j \in I_i\}) \cup X_i$ has a better function value than $\mathcal{P}$. Consequently it holds $|I_i| = 1$ for all $i = 1, \ldots, k$. This implies $\mathcal{P} = \mathcal{P}^*$, i.e., $w \in R(\mathcal{P}^*)$ and $w \notin R(\mathcal{P})$ for all Potts partitions $\mathcal{P} \neq \mathcal{P}^*$.

Let $v \in \mathbb{R}_+^{|E(G)|}$ with $v_e = 1$ for all $e \in E(G)$. By similar arguments, it is $(w + \epsilon v) \in R(\mathcal{P}^*)$ and $(w + \epsilon v) \notin R(\mathcal{P})$ for all Potts partitions $\mathcal{P} \neq \mathcal{P}^*$ if the real number $\epsilon > 0$ is sufficiently small. Additionally we have $w_e + \epsilon v_e > 0$ for all $e \in E(G)$, i.e., $(w + \epsilon v) \in \operatorname{int} R(\mathcal{P}^*)$. Consequently, $\operatorname{int} R(\mathcal{P}^*) \neq \emptyset$, and the full-dimensionality follows. □

Next we show that the regions of stability provide a partition of the set $\mathbb{R}_+^{|E(G)}$. For this, we need to prove that the regions of stability overlap only at their boundaries and that $\mathbb{R}_+^{|E(G)|}$ is covered by the latter. Finally, we show that for all overlapping regions of stability $R(\mathcal{P}_1)$ and $R(\mathcal{P}_2)$ the intersection $R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$ is a face of both $R(\mathcal{P}_1)$ and $R(\mathcal{P}_2)$.

Theorem 4.3. *1. $\mathbb{R}_+^{|E(G)|}$ is covered by the set of all regions of stability, i.e.,*

$$\mathbb{R}_+^{|E(G)|} = \bigcup_{\mathcal{P} \in \mathfrak{P}_G} R(\mathcal{P}).$$

2. Each pair of regions of stability overlap each other only at their boundary, i.e.,

$$\operatorname{int}(R(\mathcal{P}_1) \cap R(\mathcal{P}_2)) = \emptyset \quad \textit{for all } \mathcal{P}_1, \mathcal{P}_2 \in \mathfrak{P}_G,\ \mathcal{P}_1 \neq \mathcal{P}_2.$$

3. Let $\mathcal{P}_1, \mathcal{P}_2 \in \mathfrak{P}_G$ be some arbitrary Potts partitions with

$$R(\mathcal{P}_1) \cap R(\mathcal{P}_2) \neq \emptyset.$$

Then, $R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$ is a face of both polyhedrons $R(\mathcal{P}_1)$ and $R(\mathcal{P}_2)$.

Proof. 1. The first property is equivalent to $\Psi^*_{\mathfrak{P}_G}(w) \neq \emptyset$ for all $w \in \mathbb{R}_+^{|E(G)|}$. The latter is true due to the finite cardinality of $\mathfrak{P}_G$.

2. Let $\mathcal{P}_1, \mathcal{P}_2 \in \mathfrak{P}_G$, $\mathcal{P}_1 \neq \mathcal{P}_2$, be arbitrary Potts partitions. If $R(\mathcal{P}_1) \cap R(\mathcal{P}_2) = \emptyset$, there is nothing to show. Let $w \in R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$, and thus, $F_{G,w}(\mathcal{P}_1) = F_{G,w}(\mathcal{P}_2)$. Further, let $v \in \mathbb{R}^{|E(G)|}$ with

$$v_e = \begin{cases} 1 \text{ for } e \in E(\mathcal{P}_1) \backslash E(\mathcal{P}_2) \\ -1 \text{ for } e \in E(\mathcal{P}_2) \backslash E(\mathcal{P}_1) \\ 0 \qquad\qquad \text{otherwise.} \end{cases}$$

Then for all real numbers $\epsilon > 0$ we have

$$F_{G,w+\epsilon v}(\mathcal{P}_1) = F_{G,w}(\mathcal{P}_1) + \epsilon|E(\mathcal{P}_1)\backslash E(\mathcal{P}_2)|$$

and therefore

$$\begin{aligned} F_{G,w+\epsilon v}(\mathcal{P}_2) &= F_{G,w}(\mathcal{P}_2) - \epsilon|E(\mathcal{P}_2)\backslash E(\mathcal{P}_1)| \\ &= F_{G,w}(\mathcal{P}_1) - \epsilon|E(\mathcal{P}_2)\backslash E(\mathcal{P}_1)| \\ &= F_{G,w+\epsilon v}(\mathcal{P}_1) - \epsilon|E(\mathcal{P}_1)\backslash E(\mathcal{P}_2)| - \epsilon|E(\mathcal{P}_2)\backslash E(\mathcal{P}_1)|) \\ &< F_{G,w+\epsilon v}(\mathcal{P}_1) \quad \text{because of } \mathcal{P}_1 \neq \mathcal{P}_2. \end{aligned}$$

The latter implies $w + \epsilon v \notin R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$ for all real numbers $\epsilon > 0$, i.e., $w \notin \mathrm{int}(R(\mathcal{P}_1) \cap R(\mathcal{P}_2))$. Hence the second property is verified.

3. Let $\mathcal{P}_1, \mathcal{P}_2 \in \mathfrak{P}_G$, $\mathcal{P}_1 \neq \mathcal{P}_2$, be two Potts partitions with $R(\mathcal{P}_1) \cap R(\mathcal{P}_2) \neq \emptyset$. Then, obviously it is

$$\begin{aligned} R(\mathcal{P}_1) \cap R(\mathcal{P}_2) = \big\{ w \in \mathbb{R}_+^{|E(G)|} : & F_{G,w}(\mathcal{P}_1) = F_{G,w}(\mathcal{P}_2) \\ & F_{G,w}(\mathcal{P}_1) \geq F_{G,w}(\mathcal{P}) \quad \forall \mathcal{P} \in \mathfrak{P}_G \big\}. \end{aligned}$$

Thus we obtain $R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$ by replacing exactly one inequality by an equation in both formulas (4.27) for the regions of stability $R(\mathcal{P}_1)$ and $R(\mathcal{P}_2)$ (see Lemma 4.2). Hence, $R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$ is a face of both $R(\mathcal{P}_1)$ and $R(\mathcal{P}_2)$. □

Using formula (4.26) it is easy to see that $\Psi^*_{\mathfrak{P}_G}(w) = \{\mathcal{P}\}$ for all $w \in \mathrm{int}R(\mathcal{P})$. Together with Theorem 4.3 this implies that for almost all $w \in \mathbb{R}_+^{|E(G)|}$ there exists a unique optimal solution.

If we want to compute a region of stability $R(\mathcal{P}^*)$, it is not efficient to use formula (4.27) without modifications since the set $\mathfrak{P}_G$ has a large cardinality and since often there are a lot of redundant inequalities. The next theorem gives a sufficient criterion for an inequality to be redundant.

Theorem 4.4. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two incomparable Potts partitions. Then $R(\mathcal{P}_1)$ and $R(\mathcal{P}_2)$ do not have a common face of dimension $|E(G)| - 1$, i.e., $\mathcal{P}_2$ is redundant for the computation of $R(\mathcal{P}_1)$.*

Proof. In case $R(\mathcal{P}_1) \cap R(\mathcal{P}_2) = \emptyset$ the assertion obviously holds. Thus, let $w \in R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$. Then, because of Corollary 4.2 it is true that $\mathcal{P}_1 \wedge \mathcal{P}_2 \in \Psi^*_{\mathfrak{P}_G}(w)$ and $\mathcal{P}_1 \vee \mathcal{P}_2 \in \Psi^*_{\mathfrak{P}_G}(w)$. This implies $w \in R(\mathcal{P}_1 \vee \mathcal{P}_2) \cap R(\mathcal{P}_1 \wedge \mathcal{P}_2)$ for all $w \in R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$, i.e.,

$$R(\mathcal{P}_1) \cap R(\mathcal{P}_2) \subseteq R(\mathcal{P}_1 \wedge \mathcal{P}_2) \cap R(\mathcal{P}_1 \vee \mathcal{P}_2) \cap R(\mathcal{P}_1),$$

where $\mathcal{P}_1 \wedge \mathcal{P}_2$, $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_1 \vee \mathcal{P}_2$ are pairwise different because of $\mathcal{P}_1 || \mathcal{P}_2$. Since it is contained in the intersection of at least three full-dimensinal polyhedrons, the dimension of $R(\mathcal{P}_1) \cap R(\mathcal{P}_2)$ is smaller than $|E(G)| - 1$. □

5 Implementation and Experimental Results

We implemented both the original algorithm by Anglès d'Auriac et al. and our modifications within the same framework, using the OGDF library [17]. For the minimum cut computations, we used a fast implementation of the Goldberg-Tarjan algorithm [16]. The runs were performed on an Intel Celeron machine with 1.86 GHz.

Before starting Algorithm 3.2, we ran a heuristic for the computation of problem (1.1). This heuristic is constructed simply by skipping the minimum cut computations in Step 6. of Algorithm 3.2. In order to solve the problem exactly, the heuristics is followed by the improved exact algorithm. Furthermore, in Step 5. of the latter, we used fast heuristics for the cycle search and an enumerative algorithm for the $F_2 = G(W)$ subgraphs consisting of two triangles sharing an edge for which $\{W\}$ is optimal for F_2.

As we are not aware of other experimental results presented in the literature, we focused on the physics application. For the latter, instances defined on regular grid graphs in d dimensions with weights chosen according to some probability distribution are very relevant. In order to be able to compare our results with the algorithm implemented in [1], we used the same test bed. We studied two-dimensional square grid graphs with randomly chosen weights that can take only two different values, $w_1 \neq w_2$, where $p\%$ of the edges have weight w_1. Furthermore, the weights satisfy $w_1 + w_2 = 1$. We vary the size L of the $L \times L$ grid, w_1 and p. As the weights are randomly chosen, we always report averages over 20 different instances of the same class.

We compare the behaviour of the different algorithms for $L = 128, 256$ and different values of w_1 and p in Table 1 and Table 2. We report the CPU times in seconds, the number of minimum cut computations, and the maximum size of U, for which a minimum cut is computed.

p	Improved Method			Algorithm 3.1		
	# mincut comp.	maximal U	CPU time	# mincut comp.	maximal U	CPU time
80	6.30	64.55	0.48	16383	16301.80	138.06
70	210.65	1566.00	4.50	16383	15979.70	145.93
60	1765.30	9983.85	19.07	16383	14979.65	153.05
50	2364.10	5738.45	19.03	16383	7635.80	115.61
40	265.05	183.70	4.93	16383	907.30	21.43
30	3.90	13.55	3.15	16383	218.45	14.10
20	0.25	0.55	2.14	16383	133.40	12.50
80	23.95	265.05	4.64	65535	65232.10	2273.66
70	904.30	6851.65	102.08	65535	63925.50	2510.13
60	6898.85	39705.05	330.18	65535	59817.25	2772.62
50	9265.45	22285.20	340.97	65535	30089.30	2310.81
40	1047.75	671.90	86.32	65535	3400.40	401.48
30	23.30	41.70	55.01	65535	765.55	245.17
20	0.85	1.45	32.22	65535	262.85	212.49

Table 1. Results for $L = 128$ (top), $L = 256$ (bottom) and $w_1 = 0.2$. CPU times are given in seconds.

For the choice of $w_1 = 0.2$, the cycle search heuristics is very often successful. Moreover, the heuristic for large p often solves the problem exactly. Also for smaller p the heuristic helps reducing the running time. We note that the basic Algorithm 3.1 always computes $|V| - 1$ minimum cuts, independently of the distribution of the edge weights. In contrast, in the improved algorithm the number depends on the choice of w_1 and p.

For $w_1 = 0.4$, the heuristic has only a running time of about 0.2 seconds, but it is never successful in reducing the size of the graph. Furthermore, the cycle search algorithms contribute less than for $w_1 = 0.2$. However, for $w_1 = 0.4$ the search for the subgraphs F_2 and Step 4. of Algorithm 3.2 are often successful. We compare the running times in seconds in Table 2. Clearly, the performance can drastically be improved by the above-mentioned graph-theoretic considerations.

	L	p=80	p=70	p=60	p=50	p=40	p=30	p=20
Impr. Method	128	9.24	17.26	28.55	74.95	8.09	3.38	2.89
Algorithm 3.1	128	142.64	149.96	150.52	279.99	15.08	13.09	12.82
Impr. Method	256	118.55	252.29	473.60	2116.18	81.03	32.09	32.73
Algorithm 3.1	256	2248.08	2418.49	2614.13	8454.78	219.73	208.77	207.20

Table 2. Solution times in seconds for $w_1 = 0.4$

In 2002, Anglès d'Auriac et al. presented average running times for $L \times L$ grid graphs with $w_1 + w_2 = 1$, using a Pentium III processor with 800 MHz. Their program needed on average 1.5 hours for a 128^2 grid and one day of CPU time for a grid of size 256^2. Our implementation of the basic algorithm needs roughly half an hour on average for $L = 256$, whereas the improved method only takes ca. 6 minutes. We note that the machine we used is considerably faster, and so we cannot easily compare our running times with those reported in [1].

6 Conclusions

In this work, we presented a fast algorithm for the problem of optimum cooperation. By an intensive study of the underlying graph-theoretic problem, the running time of the solution algorithm can considerably be reduced. Finally, we analyzed the structure of the solution sets.

Acknowledgments

We are grateful to Andrea Wagner for significant help with the implementation of the algorithm. We thank Heiko Rieger for valuable remarks on the physics aspects of the problem.

References

1. Anglès d'Auriac JCh, Iglói F, Preissmann M, and Sebö A (2002) Optimal cooperation and submodularity for computing Potts' partition functions with a large number of states. J Phys A: Math Gen 35:6973-6983
2. Baïou M, Barahona F, Mahjoub R (2000) Separation of partition inequalities. Math of Operations Research, 25(2):243-254
3. Barahona F (1992) Separating from the dominant of the spanning tree polytope. Operations Research Letters 12:201-203
4. McCormick ST (2005) Submodular Function Minimization. In: Aardal K et al (ed) Discrete Optimization: Handbooks in Operations Research and Management Science Vol.12 , Elsevier, Amsterdam, pp 321-391
5. Cunningham WH (1985) Optimal Attack and Reinforcement of a Network. Journal of the Association for Computing Machinery 32(3):549-561
6. Dempe S, Schreier H (2006) Operations Research - Deterministische Modelle und Methoden. Teubner-Verlag, Wiesbaden
7. Fujishige S (2005) Submodular Functions and Optimization. Annals of Discrete Mathematics Vol.58, Elsevier, Amsterdam
8. Goldberg AV, Tarjan RE (1988) A new approach to the maximum-flow problem. Journal of the ACM 35(4):921-940.
9. Grätzer G (1978) General Lattice Theory. Birkhäuser-Verlag, Basel
10. Grötschel M, Lovász L, Schrijver A (1981) The ellipsoid method and its consequences in combinatorial optimization. Combinatorica 1(2):169-197
11. Grötschel M, Lovász L, Schrijver A (1988) Geometric Algorithms and Combinatorial Optimization. Springer Verlag
12. Hartmann AK, Rieger H (2002) Optimization Algorithms in Physics. Wiley-VHC, Berlin
13. Hartmann AK, Rieger H (2004) New Optimization Algorithms in Physics, Wiley-VHC, Berlin
14. Iwata S, Fleischer L, and Fujishige S (2001) A combinatorial strongly polynomial algorithm for minimizing submodular functions. Journal of the ACM 48(4):761-777
15. Juhasz R, Rieger H, Iglói F (2001) The random-bond Potts model in the large-q limit. Phys Rev E 64:056122
16. Jünger M, Rinaldi G, Thienel S (2000) Practical Performance of Efficient Minimum Cut Algorithms. Algorithmica 26:172-195
17. Open Graph Drawing Framework, `www.ogdf.net`
18. Preissmann M, Sebö A Graphic submodular function minimization: an graphic approach and applications , unpublished
19. Schrijver A (2000) A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time. J Combinatorial Theory B 80:346-355
20. Topkis DM (1998) Supermodularity and Complementarity. Princeton University Press, Princeton