Lagrangian Smoothing Heuristics for Max-Cut†

Hernán Alperin‡‡ Ivo Nowak*

May 28, 2002

Humboldt-Universität zu Berlin
Institut für Mathematik
Rudower Chaussee 25,
D-12489 Berlin, Germany

Abstract

This paper presents smoothing heuristics for an NP-hard combinatorial problem based on Lagrangian relaxation. We formulate the Lagrangian dual for this nonconvex quadratic problem and propose eigenvalue nonsmooth unconstrained optimization to solve the dual problem with bundle or subgradient methods. Derived heuristics are considered to obtain good primal solutions through pathfollowing methods using a projected gradient algorithm. Starting points are drawn using several sampling techniques that use randomization and eigenvectors.

The proposed method turns out to be competitive with the most recent ones. The idea presented here is generic and can be generalized, to all problems where convex Lagrangian relaxation can be applied. Furthermore, to the best of our knowledge, this is the first time that a Lagrangian heuristic is combined with pathfollowing techniques.

Key words. semidefinite programming, quadratic programming, combinatorial optimization, non-convex programming, approximation methods and heuristics, pathfollowing, homotopy

AMS classifications. 90C22, 90C20, 90C27, 90C26, 90C59

†The work was supported by the German Research Foundation (DFG) under grant NO 421/2-1.
‡‡`alpe@mathematik.hu-berlin.de`
*`ivo@mathematik.hu-berlin.de`

1 Introduction

Let $G = (N, E)$ be an undirected weighted graph consisting of the set of nodes N and the set of arcs or edges E. Let a_{ij} be the cost of edge ij, and assume G complete, otherwise set $a_{ij} = 0$ for every edge ij not in E. The Maximum Cut problem consists of finding a subset of the nodes, $S \subset N$ that maximizes the cut function

$$\mathrm{cut}(S) = \sum_{ij \in \delta(S)} a_{ij},$$

where the incidence function $\delta(S) = \{ij \in E \text{ such that } i \in S \text{ and } j \notin S\}$, is defined to be the set of arcs that cross the boundary of S.

Let the vector x, with $x_i \in \{-1, 1\}$, represent whether $i \in S$. In this way, $ij \in \delta(S)$, i.e. nodes i and j are on different sides of the cut, if and only if $x_i x_j < 0$. We can write the maximum cut problem as a quadratic nonconvex problem,

$$\text{(MC-Q)} \qquad \begin{array}{ll} \min & x^T A x \\ \text{s.t.} & Xx = e \end{array}$$

where $X = \mathrm{Diag}(x)$ is the matrix with the vector x in its diagonal and zeros elsewhere, and $e \in I\!R^n$ is the vector of ones. We can observe that $Xx = e$ if and only if $x \in \{-1, 1\}^n$ and

$$x^T A x = \sum_{\substack{i,j=1 \\ x_i x_j > 0}}^{n} a_{ij} - \sum_{\substack{i,j=1 \\ x_i x_j < 0}}^{n} a_{ij} = \sum_{i,j=1}^{n} a_{ij} - 2 \sum_{\substack{i,j=1 \\ x_i x_j < 0}}^{n} a_{ij} = e^T A e - 4 \sum_{ij \in \delta(S)} a_{ij}.$$

Therefore, the maximum $\mathrm{cut}(S)$ can be produced by minimizing $x^T A x$, then adding the constant $e^T A e$, and finally dividing by 4.

Regardless that problem (MC-Q) was proved to be NP-hard [9], some interesting heuristics to obtain good solutions have been proposed. Following, we present the most known and recent ones.

Let $Z \succcurlyeq 0$ be a semidefinite positive matrix in $I\!R^{(n,n)}$, $\mathrm{diag}(Z)$ be the vector representation of the diagonal of the matrix Z, and $A \bullet B = \mathrm{trace}(AB) = \sum_{ij} a_{ij} b_{ij}$, be the Hadamark product. Problem (MC-Q) can be also written as

$$\text{(MC-SDP)} \qquad \begin{array}{lr} \min & A \bullet Z \\ \text{s.t.} & \mathrm{diag}(Z) = e \\ & Z = x x^T \\ & Z \succcurlyeq 0, \end{array}$$

by noting that the constraint $Z = xx^T$ implies that $\text{diag}(Z) = Xx$, and that $\text{trace}(AZ) = \text{trace}(Axx^T) = x^T Ax$. By removing the rank-1 constraint, we have the semidefinite relaxation problem

$$\text{(SDP)} \qquad \begin{array}{ll} \min & A \bullet Z \\ \text{s.t.} & \text{diag}(Z) = e \\ & Z \succcurlyeq 0. \end{array}$$

Goemans and Williamson [10] used the solution of (SDP) to generate random feasible points $\hat{x}$. Assuming that Z^* is a (SDP) optimal solution, which is not necessarily rank-1, their strategy consists of finding a factorization $Z^* = V^T V$, where V can be the Choleski factorization. A feasible solution $\hat{x} = \text{sign}(V^T u)$ can be produced using the random vector $u \sim U[B(0,1)]$, uniformly distributed over the zero centered n dimensional ball of radius one. For the case of non-negative edges weight, $a_{ij} \geq 0$, they proved a bound on the expected value of the randomly generated solutions that is

$$E(\text{cut}(\hat{x})) \geq .878\,\text{val(MC-Q)},$$

where val(MC-Q) is the optimal value of problem (MC-Q), and $\text{cut}(\hat{x})$ is $\text{cut}(S)$ for the set S defined by $\hat{x}$.

Later Bertsimas and Ye [4] proved that it is possible to use the SDP optimal solution, Z^*, as covariance matrix to generate a vector $x \sim N(0, Z^*)$. This randomly generated vector can be used to produce a feasible one, $\hat{x} = \text{sign}(x)$, leading to the same results.

Unfortunately, solving problem (SDP) by means of a primal-dual interior point algorithm can require quite a long time if the problem contains thousands of variables. Besides, the sparsity of the matrix is lost when the problem is solved by that method. Several SDP methods exploiting sparsity have been proposed, such as the purely dual interior point algorithm [2], nonlinear programming approach [6], and the spectral bundle method [15]. Burer et. al. [5] have devised a rank-2 relaxation of problem (MC-Q). In their heuristic, they relax the binary vector into a vector of angles, and work with an angular representation of the cut. They maximize an unconstrained sigmoidal function to obtain heuristic points, that later are perturbed to improve the results of the algorithm. Their approach is similar to the Lorena [3] algorithm.

The Max-Cut problem can also be formulated as an unconstrained quadratic binary problem, (UQB) [14, 19]. Metaheuristics for solving UQB are proposed and studied for cases containing up to 2500 variables [1].

In this paper, we propose a pathfollowing heuristic, which is also called homotopy, deformation, continuation, or smoothing heuristic. The heuristic is based on a parametric optimization problem defined as a convex combination between a Lagrangian relaxation and the original problem. Starting from the Lagrangian relaxation, a path-following method is applied to obtain good solutions while gradually transforming the relaxed problem into the original problem formulated with an exact penalty function.

Paths of this parametric optimization problem are traced using a projected gradient method and the final points are rounded. To follow this idea, starting points for the procedure are needed. Hence, We present different sampling techniques to generate the initial points.

This method aims to avoid regions with less interesting local optima, and direct towards regions where the global optimum is likely to be. Such kind of heuristics have been applied to energy optimization problems, where the relaxation is obtained using Gaussian transformations. Scheltstraete et al. [21] provide an overview on this kind of heuristics.

Feltenmark and Kiwiel [8] proposed a Lagrangian heuristic which can be applied to very general optimization problems. In that paper, they observed that higher dual objective accuracy need not necessarily imply better quality of the heuristic primal solution. Our results from the sampling spaces also pose questions regarding the importance of solving the dual problem until optimality. We obtain good solutions with our heuristic by using feasible dual points that are not necessary close to optimality. In many cases, eigenvalue corrected dual points already yield good results.

To the best of our knowledge, it is the first time that a Lagrangian heuristic is combined with smoothing techniques. Since the approach is generic we believe that it can be generalized to any optimization problems where convex Lagrangian relaxation can be applied.

The paper is organized in six sections. The next section introduces the Lagrangian convexification of problem (MC-Q) and explains the dual formulation. Section 3 defines the pathfollowing heuristic and describe its parameters, and section 4 explains the sampling methods for generating starting points for the algorithm. Finally, sections 5 presents the results and our conclusions are presented in section 6.

2 Lagrangian Convexification

In this section we describe how relaxations of problem (MC-Q) can be obtained using duality theory. We first formulate the Lagrangian dual, then transform it to its eigenvalue formulation, and describe how it can be optimized using bundle or other subgradient methods for nonsmooth optimization.

2.1 Lagrangian dual

The Lagrangian function of problem (MC-Q) can be written as

$$L\,(x;\mu) = -e^T\mu + x^T(A+M)x,$$

where $M = \mathrm{Diag}(\mu)$, $\mu \in I\!R^n$ is a vector of Lagrangian multipliers. The dual function,

$$D(\mu) = \inf_{x\in I\!R^n} L\,(x;\mu)\,,$$

can be clearly written in close form as

$$D(\mu) = \begin{cases} -e^T\mu & \text{if } A+M \succcurlyeq 0 \\ -\infty & \text{if } A+M \not\succcurlyeq 0. \end{cases} \tag{1}$$

Finally, the Lagrangian dual problem of (MC-Q) is

$$\text{(D)} \qquad \sup_{\mu\in I\!R^n} D(\mu).$$

The close form (1) shows that for each $\mu \in \operatorname{dom} D$ the Lagrangian $L(\cdot;\mu)$ is a convex underestimator of the objective function over $[-e,e]$. Note also that (D) is equivalent to the semidefinite program

$$\text{(D-SDP)} \qquad \begin{array}{ll} \max & -e^T\mu \\ \text{s.t.} & A+M \succcurlyeq 0, \end{array}$$

which is the dual of (SDP). The related duality gap is zero since the Slater condition holds [14].

Remark 1 *It is interesting to note that problem (D-SDP) can be also formulated as finding the minimum sum of elements to set in the diagonal of A such that it becomes positive semidefinite,* $\min \sum \mu_i$ *such that* $A+M \succcurlyeq 0$.

Lemma 1 *The optimum of problem (D-SDP) is attained when the smallest eigenvalue of $A+M$ is zero.*

Proof. Let us assume that $A+M \succ 0$. This means that all its eigenvalues are positive. Let $\omega > 0$ be the smallest of the eigenvalues of $A+M$. We can define $M^* = M - \omega I$ with $A + M^* = A + M - \omega I \succcurlyeq 0$. Hence, $e^T\mu^* = e^T\mu - n\omega < e^T\mu$, showing that μ could not have been the optimum. $\square$

2.2 Eigenvalue formulation

Let us now formulate the dual problem (D) as an eigenvalue optimization problem. Consider the sphere

$$S = \{x \in I\!R^n \mid \|x\|^2 = n\},$$

which clearly is the only sphere that contains the feasible set, $\{-1,1\}^n \subset S$.

Lemma 2 *Let $\lambda_1(A+M)$ denote the smallest eigenvalue of $A+M$. Then, the following equality holds*

$$\lambda_1(A+M) = \min_{\|x\|=1} x^T(A+M)x,$$

Proof. Let Λ be the diagonal matrix of eigenvalues of $A+M$, let U be the base of eigenvectors, in columnwise matrix form, and let $\xi = Ux$, be the spectral representation of x, i.e. the writing of x in the eigenvector space. Then, the variable objective part of the Lagrangian, $x^T(A+M)x = x^TU^T\Lambda Ux = \xi^T\Lambda\xi = \sum \lambda_i\xi_i^2$, is minimized when $\xi_1 = 1$ and $\xi_i = 0$ for $i \neq 1$, given that λ_1 is the smallest eigenvalue, and $\|\xi\| = \|x\|$ since the matrix U is orthonormal. $\square$

We define the dual function with respect to S,

$$D_S(\mu) = \min_{x\in S} L(x;\mu) = -e^T\mu + n\lambda_1(A+M), \tag{2}$$

and the corresponding dual problem is therefore

$$(\mathrm{D}_S) \qquad \sup_{\mu\in I\!R^n} D_S(\mu) = \sup_{\mu\in I\!R^n} -e^T\mu + n\lambda_1(A+M).$$

We can state the following relationships between solutions of (D) and (D_S).

Proposition 1
(i) It holds that $D(\mu - \lambda_1(A+M)e) = D_S(\mu)$*.*
(ii) The optimum values of (D) and (D_S) *are the same.*
(iii) If μ *is a solution of* (D_S) *then* $\mu^* = \mu - \lambda_1(A+M)e$ *is a solution of (D).*

Proof.
(i) Given $\omega = \lambda_1(A+M)$, the matrix $A + M - \omega I$ is clearly positive semidefinite. From the close form (1) of the dual function, it follows $D(\mu - \omega e) = -e^T(\mu - \omega e) = -e^T\mu + n\omega = D_S(\mu)$.
(ii) From Lemma 1, a solution μ of (D) fulfils $\lambda_1(A+M) = 0$. Hence, val(D) = val(D_S) follows from (i).
(iii) follows from (i) and (ii). □

Remark 2 *The transformation* $\hat{\mu} = \mu - \lambda_1(A+M)e$ *maps an arbitrary* $\mu \in I\!R^n$ *onto* $\operatorname{dom} D$*, the domain of D. Thus, the Lagrangian* $L(x; \hat{\mu})$ *is a convex underestimating function.*

2.3 Supergradient formula

A supergradient of a non-necessarily differentiable concave function $D : I\!R^m \to I\!R$, is a vector $g \in I\!R^m$, satisfying

$$D(\mu) + g^T(\lambda - \mu) \geq D(\lambda)$$

for all $\lambda, \mu \in I\!R^m$. The previous definition is widely known in its reverse form for convex functions and subgradients. A supergradient of a dual function can be computed by evaluating the constraint functions at a Lagrangian solution point. The following Lemma holds [16]:

Lemma 3 *Let* $L(x; \mu) = f(x) + \mu^T g$ *be a continuous Lagrangian function related to an objective function* $f : I\!R^n \to I\!R$ *and a constraint function* $h : I\!R^n \to I\!R^m$*. Let* $X \subset I\!R^n$ *be a compact set. Then the dual function* $D(\mu)$ *is concave and* $g(\mu) = h(x_\mu)$ *is a supergradient of* $D(\mu)$ *at* $\mu \in \operatorname{dom}(D)$*, where* x_μ *is a Lagrangian solution of* $D(\mu)$*, i.e.*

$$x_\mu \in \operatorname*{Argmin}_{x \in X} L(x; \mu).$$

The previous result can be applied to solve the unconstrained problem (D_S) using subgradient optimization techniques.

Proposition 2 *For a given dual point $\mu \in I\!R^n$, let $v \in I\!R^n$ be a minimum eigenvector of $A+M$, with $\|v\| = 1$, and let $x_\mu = n^{1/2} v$ be a solution of the Lagrangian problem (2). Then $g \in I\!R^n$, defined by $g = X_\mu x_\mu - e$, is a supergradient of $D_S(\mu)$ at μ.*

Proof. The statement follows directly from Lemma 3. $\square$

3 Pathfollowing Heuristics

A pathfollowing method works by first solving a simple problem and then deforming this problem into the complicated original one. During this deformation or homotopy some (or all) paths from the solution of the simple problem to the solutions of the original problem are followed. Based on this idea, we present now a heuristic for problem (MC-Q).

3.1 Homotopy

Let us define the box constrained quadratic penalty parametric problem,

$$(\text{MC-P}_\gamma) \qquad \begin{array}{ll} \min & x^T A x + \gamma(n - \|x\|^2) \\ \text{s.t.} & x \in [-e, e], \end{array}$$

which is clearly equivalent to (MC-Q) for γ sufficiently large.

By setting $\gamma(t) = (1-t)^{-1}$, we ensure that γ is large enough when t is sufficiently close to 1. Therefore, $P(x;t) = x^T A x + (1-t)^{-1}(n - \|x\|^2)$ is an exact penalty function for $x \in [-e, e]$ in the unit box. Let $L(x;\mu) = -e^T\mu + x^T(A+M)x$ be the Lagrangian function regarding the binary constraints. We can now define a homotopy function with parameter t, between the penalty objective function of problem (MC-P$_\gamma$), $P(x;t)$, and the Lagrangian $L(x;\mu)$, to be

$$\begin{aligned} H(x;\mu,t) &= tP(x;t) + (1-t)L(x;\mu) \\ &= t(1-t)^{-1}n - e^T\mu + x^T(A - t(1-t)^{-1}I + (1-t)M)x. \end{aligned} \qquad (3)$$

The formulation (3) shows that the function H is quadratic on x. Finally, we consider the following parametric optimization problem associated to the homotopy function,

$$(\text{P}_t) \qquad \min_{x \in [-e,e]} H(x;\mu,t).$$

Assuming that $\mu \in \operatorname{dom} D$, the function $H(\cdot;\mu,0) = L(\cdot;\mu)$ is a convex underestimator of (MC-Q) and (P$_0$) is a convex optimization problem. On the other hand, when t tends to 1, $H(\cdot;\cdot,t)$ tends to $P(\cdot;t)$, which becomes concave for t close to 1. The latter motivates the following Lemma.

Lemma 4
(i) There exists $t^0 \in (0,1)$ such that $\mathrm{val}(MC\text{-}Q) = \mathrm{val}(P_t)$ *for all* $t \in [t^0, 1)$.
(ii) The function $v(t) = \mathrm{val}(P_t)$ *is continuous on the interval* $[0, t^0]$.

Proof. (i) It is clear that there exist $t^0 \in (0,1)$ such that $H(\cdot;\mu,t)$ is concave for $t \in [t^0, 1)$. To see that, consider the Hessian of the homotopy, from the formulation (3), $\nabla^2 H = A - t\gamma I + (1-t)M$. Since $\gamma \to \infty$ when $t \to 1$, the matrix $\nabla^2 H$ is negative definite for some $t^0 < 1$. Furthermore, $H(x;\mu,t) = x^T A x$ for all $x \in \{-1,1\}^n$, for all $\mu \in I\!R^n$, and $t \in I\!R$, which proves the statement.
(ii) This fact is clear from the continuity of P and L since v is a projection of $I\!R^n$ onto $[0,1)$. $\square$

Remark 3 *Note that it might be possible that the path $x(t)$ of the parametric optimization problem (P_t) related to a solution x^* of (MC-Q) is discontinuous, in other words there might be jumps [13].*

Remark 4 *Dentcheva et al. [7] and Guddat et al. [12] pointed out general disadvantages of this formulations, i. e. the one-parameter optimization is not defined for $t = 1$ and the objective function could be only once continuously differentiable. However, for the Max-Cut problem the penalty objective function used is quadratic, thus infinitely many times differentiable since no inequality constraints were used. On the other side, from Lemma 4, the path need not to be traced until $t = 1$, disregarding that undefined point.*

3.2 Pathfollowing Algorithm

The solution of the parametric optimization problem (P_t) is as difficult as solving (MC-Q). Therefore, we trace approximately a path of (P_t) using a truncated projected gradient algorithm for approximating a solution of $\min_{x\in[-e,e]} H(x;\mu,t_k)$.

Let $\Pi_{[-e,e]}(x)$ be the *box projector* operator

$$\Pi_{[-e,e]}(x)_i = \begin{cases} -1 & \text{if} \quad x_i < -1 \\ x_i & \text{if} \quad -1 \le x_i \le 1 \\ 1 & \text{if} \quad x_i > 1. \end{cases}$$

Figure 1 presents Algorithm `pathfollowing` that traces a single path towards a local optima. The Algorithm depends highly on the initial points, which are provided by sampling techniques explained in section 4, and certain parameters explained in the following subsection.

Algorithm $\hat{x} =$ `pathfollowing`(x, μ)

```
set     k := 0;
        t_0 := 0;
        x^0 := x;
do      get t_{k+1} > t_k;
        set y^0 := x^k;
        for j := 0 to m − 1 do
              y^{j+1} := Π_{[−e,e]} ( y^j − β^j ∇H(y^j; μ, t_k) / ‖∇H(y^j; μ, t_k)‖ );
        set k := k + 1;
              x^k = y^m;
until stopping criteria fulfilled or k = M;
return x̂ = sign(x^k).
```

$$y^{j+1} := \Pi_{[-e,e]}\left(y^j - \beta^j \frac{\nabla H(y^j;\mu,t_k)}{\|\nabla H(y^j;\mu,t_k)\|}\right);$$

Figure 1: Algorithm `pathfollowing` traces a path from a starting primal point x to a feasible point $\hat{x}$. The parameters are discussed in subsection 3.3. The inner loop solves approximately $x^k = \operatorname{argmin}_{x\in[-e,e]} H(x;\mu,t_k)$ by making m steps (minor iterations) of a projected gradient algorithm.

3.3 Parameters

Algorithm `pathfollowing` assumes that the following parameters are provided.

major iterations. The parameter M controls the maximum number the outer loop is performed.

minor iterations. The parameter m controls the number of iterations for the projected gradient algorithm in the inner loop.

steplength. The parameters β^j, for $j = 1, \ldots, m$, determine the steplength of the projected gradient algorithm. It is possible to autotune β^j to guarantee descent step using bisection rule, or to use a fixed value $\beta^j = \beta$.

homotopy sequence. The parameters $t_1 < \cdots < t_k < \cdots < t_M$, determine the values at which the function $H(x;\mu,t_k)$ is optimized. It is possible to generate t_k, using a *geometric* sequence, i.e. $t_k = 1 - \rho^k$ with $\rho \in (0,1)$, or using a *uniform* sequence, i.e. $t_k = k/(M+1)$.

3.4 Stopping Criteria

A consequence of Lemma 4 is the following proposition.

Proposition 3 *If m is large enough, Algorithm* `pathfollowing` *can be stopped if $t_k \geq t^0$ without changing the final result, where t^0 is defined in the proof of Lemma 4 (i).*

Proof. If $t_k \geq t^0$ from Lemma 4, then $H(\cdot;\mu,t_k)$ is concave. Clearly, in this case, the projected gradient algorithm converges in finitely many steps to a vertex. □

Let x^* be the global optimum of problem (MC-Q). We define the *region of attraction* to be the set of points x such that $\text{sign}(x) = \text{sign}(x^*)$. With the previous proposition, it is not needed to follow the path until the end, since after $t_k > t^0$ with m large enough, x^k is in a region of attraction and its projection will not change for larger k.

Therefore, we can use $\text{sign}(x^k) = \text{sign}(x^{k-1})$ as stopping criteria for the outer loop of the algorithm. However, this does not guarantee that $t_k > t^0$. A more careful approach is to use $\text{sign}(x^k) = \text{sign}(x^{k-1}) = \cdots = \text{sign}(x^{k-p})$, with $p > 1$.

4 Sampling

Algorithm `pathfollowing` requires initial dual and primal points. Since the algorithm is a heuristic, better results are not necessarily achieved by better dual points and their corresponding Lagrangian solutions. Therefore, we devised several techniques for generating sample points in the primal and dual space. The following subsections describe how a single dual vector μ and its corresponding primal point x are generated in each sampling method. We define two kinds of sampling methods, the first over the primal space using random techniques on different spaces, and the second over the dual space, using a sequence from an optimization algorithm. The sampling is repeated up to complete the defined sample size.

A sample of size p to start Algorithm `pathfollowing` can be represented as a set of pairs of primal and dual points, $S = \{(x^i, \mu^i) \text{ with } i = 1, \ldots, p\}$. The primal sampling sets are those whose dual points, $\mu_i = \mu$, are the same through the sample. Respectively, the dual sampling sets are those whose dual points vary through the sample or some dual method was used to obtain them. We start describing first sampling on the primal space, and we follow with the sampling in the dual space.

4.1 Random primal

A starting primal point is chosen with uniform distribution over the ball $B(0, n^{1/2})$. Then, Algorithm `pathfollowing` is applied with $\mu = -\lambda_1(A)e$. This is the simplest of all the sampling, since it does not compute the primal from dual information.

The random primal sample is therefore,

$$S_{\mathrm{RP}} = \{(x^i, \mu) : i = 1, \ldots, p,\ x^i \sim U[B(0, n^{1/2})],\ \mu = -\lambda_1(A)e\},$$

where $x^i \sim U[S]$, means that the sample point x^i is independently drawn with uniform distribution over the set S, i.e. the probability density function is the same at any point in the set, for all $y, z \in S$, we have $f_{x^i}(y) = f_{x^i}(z)$, and for all $i \neq j$, the joint probability density function equals the product of the probability density functions, $f_{x^i,x^j}(y, z) = f_{x^i}(y) f_{x^j}(z)$.

4.2 Preswitching

This is also a primal sampling, since no dual sequence is used, i.e. we do not attempt to solve the dual when generating this sample. We use the eigenvector corresponding to the smallest eigenvalue of A, $v_1(A) = v_1(A + M)$, where $\mu = -\lambda_1(A)e$, is the map into the semidefinite cone.

We define a threshold ϵ for all components of the primal point. If $x_i < \epsilon$, we multiply its value by -1 with a probability one half. This is done to explore the opposite direction when a component of the primal starting point is close to zero.

The preswitching sample is then,

$$S_{\mathrm{P}} = \{(x^i, \mu) : i = 1, \dots, p,\ x^i = \rho\chi_\epsilon(v_1(A)),\ \mu = -\lambda_1(A)e\},$$

where $\rho = n^{1/2}/\|v_1(A)\|$, and $\chi_\epsilon(v)$ is a random vector defined as

$$\chi_\epsilon(x)_j = \begin{cases} x_j & \text{if } |x_j| \geq \epsilon \\ u_j x_j & \text{if } |x_j| < \epsilon \end{cases},$$

and u_j are independent identically distributed Bernoulli random variables, which take values in $\{-1, 1\}$ with probability one half each.

4.3 Eigenspace sampling

If the duality gap is zero, an optimum primal solution lies in the kernel of the eigenspace of the minimum eigenvalue. Motivated by this fact, we generate random points in the space spanned by the eigenvectors that correspond to a certain number of smallest eigenvalues.

In particular, we define

$$S_{\mathrm{E}} = \{(x^i, \mu) : i = 1, \dots, p,\ x^i = n^{1/2} y^i/\|y^i\|,\ \mu = -\lambda_1(A)e\},$$

where

$$y^i = \sum_{k=1}^{r} \alpha_k v_k(A + M),$$

and $\alpha_k \sim N(0, 1)$, independent normally distributed, and $v_i(\cdot)$ is the eigenvector corresponding to the i-th lowest eigenvalue. The resulting random linear combination of eigenvectors, y^i is projected onto $B(0, n^{1/2})$, the ball that contains the $[-e, e]$ box.

This sampling can be combined to any sampling on the dual space by having different dual points μ^i for $i = 1, \ldots, p$, or just by selecting a single different dual point. We explain this in section 4.6 on the following page.

4.4 Random duals

This and the following, are sampling using different dual points.

To check the importance of solving the dual problem to generate good heuristic primal solutions, we generated independent random points normally distributed in the dual space, $\nu^i \sim N(0, \sigma I)$, independent identically distributed. Then, we constructed the sample correcting the dual points as explained in Remark 2,

$$S_{\mathrm{RD}} = \{(x^i, \mu^i) : i = 1, \ldots, p,\ x^i = \rho^i v_1(A + M^i),\ \mu^i = \nu^i - \lambda_1(A + N^i)e\},$$

where $\rho^i = n^{1/2}/\|v_1(A + M^i)\|$, and $N = \mathrm{Diag}(\nu)$. We map the dual points onto the semidefinite cone by subtracting the lowest eigenvalue times the vector of ones, $\mu^i = \nu^i - \lambda_1(A + N^i)e$, to ensure the convexity of the Lagrangian.

Another possible way to guarantee the convexity of $L(x; \mu)$, is to use Gershgorin sufficient condition of semidefinite positiveness, $A + M \succcurlyeq 0$ implies

$$\mu_i \geq \sum_{i \neq j} |a_{ij}|$$

since a_{ii} is zero in the Max-Cut case. However, we believe eigenvalue convexification is better. On the other side, the latter method is faster.

4.5 Dual sequence

A nonsmooth optimization method is applied on the unconstrained dual problem (D_S). This can range from simple subgradient methods, using steepest descent or conjugate subgradients, to proximal bundle method. Let $\{\nu^j\}$, for $j = 0, 1, \ldots,$ be a sequence that converges towards the optimum dual value μ^*. The dual sequence sample can be described as

$$S_{\mathrm{SD}} = \{(x^i, \mu^i) : i = 1, \ldots, p,\ x^i = \rho^i v_1(A + M^i),\ \mu^i \in \{\nu^j\}_{j=0,1,\ldots}\},$$

where $\rho^i = n^{1/2}/\|v_1(A + M^i)\|$. Each sample dual μ^i is an iterate of the dual optimization method towards obtaining the optimum dual μ^*. We pick a sample point every K iterations of the dual, i.e. $\mu^i = \nu^{iK}$.

4.6 Eigenspace after dual termination

In this last sampling, we produce starting primal points in the eigenspace of the same dual point. But we generate the dual point by optimizing the dual problem until a convergence criterion is fulfilled, $\|\mu^k - \mu^*\| < \epsilon$. Since μ^* is a priori not known, the algorithm stops if the improvement in the last r iterations was less than a fraction of the improvement made in the starting r iterations. In particular, $\|\mu^k - \mu^{k-r}\| < \theta\|\mu^r - \mu^0\|$.

The sample can be defined as

$$S_{\mathrm{ED}} = \{(x^i, \mu) : i = 1, \ldots, p,\ x^i = \rho^i y^i,\ \mu = \mu^* - \lambda_1(A + M^*)e\},$$

where $\rho^i = n^{1/2}/\|y^i\|$, $y^i = \sum_{k=1}^r \alpha_k v_k(A + M^*)$, and μ^* an optimum dual, or in fact, sufficiently close to it. The primal points are sampled from the space generated by the eigenvectors corresponding to the smallest eigenvectors of $A + M^*$, explained in section 4.3 on page 14.

5 Numerical Results

Algorithm `pathfollowing` was coded in `C++` and compiled with `g++`, the GNU compiler. Supergradients for the dual function explained in section 2 were computed according to Lemma 2. For the computation of the minimum eigenvalue and corresponding eigenvector we used the Lanczos method `ARPACK++` [11]. For solving the dual problem, we used Kiwiel's proximal bundle algorithm `NOA 3.0` [17, 18]. Pseudorandom numbers were generated using `RANLIB` library routines to simulate the proposed distributions.

The algorithm was tested using a set of examples from the 7th DIMACS Implementation Challenge [20], and using several instances created with `rudy`, a machine independent graph generator written by G. Rinaldi, which is standard for maximum cut problem [15].

The tests were run on a machine that has two 700MHz Pentium III processors and 1Gb RAM. The sampling size for all the sample sets was set to 10, and the best result over each sample type was reported.

Table 1 and Table 2 show the results for the different sampling techniques. The first reports the computing time and the second, the value in percentage refereed to the most elaborated sample S_{ED}, eigenspace after dual termination, for which the absolute result is reported. For the reported runs, we used a fixed number of major iterations M, a fixed steplength β, and a uniform sequence t_k, explained in section 3.3 on page 11.

Other combinations of the parameters described in section 3.3 on page 11 were tried. In particular, we experimented with modifications of the steplength update rule to perform line search, the sequence of values for t (a geometric sequence for $t_k = 1 - \rho^k$ was also tested), and the outer loop stopping rule explained in section 3.4 on page 12. However, we did not observed significant differences in the results. One exception was the use of simplest stopping rule criterion. In that particular case, for few primal sample examples, the algorithm stopped before achieving as good results as reported.

Previous evidence with other Lagrangian heuristics for the unit commitment problem suggests that higher dual objective accuracy need not necessarily imply better quality of the heuristic primal solution [8]. To evaluate the importance of the information provided by the dual for the heuristic, we plot comparatively the dual sequence and it corresponding heuristic primal solution sequence for some graph examples in Figure 2.

We observe that meanwhile the dual improves its value, reducing the duality gap, the

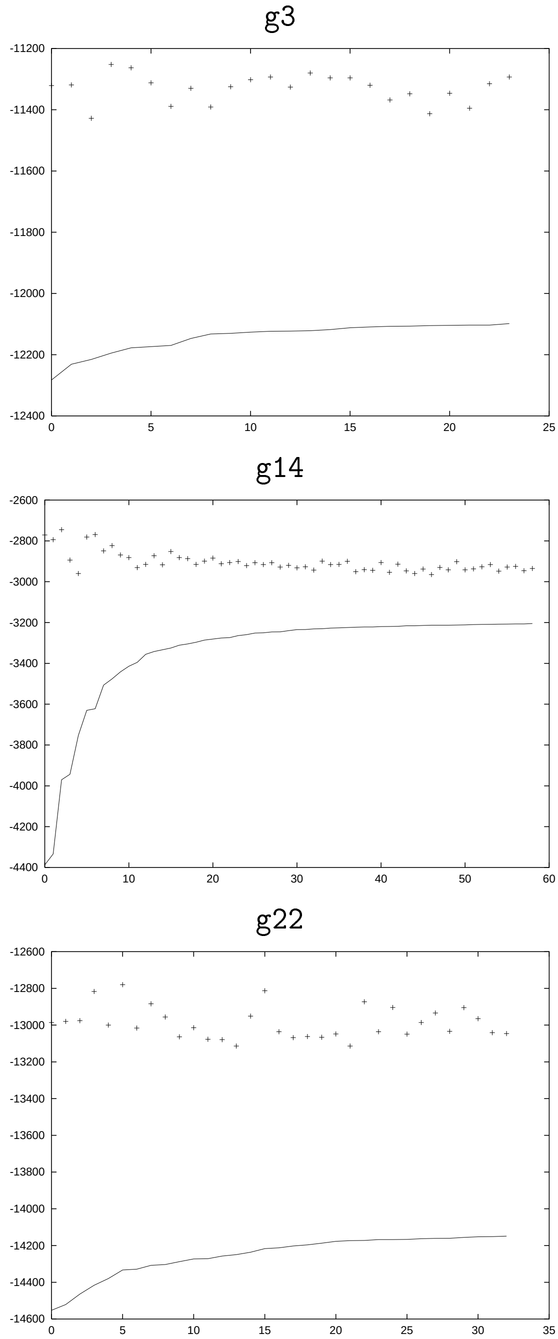
example	size		primal sampling			dual sampling		
name	n	m	S_{RP}	S_{P}	S_{E}	S_{RD}	S_{SD}	S_{ED}
g3	800	19176	15	15	17	19	1:20	32
g6	800	19176	14	14	16	16	49	1:13
g13	800	1600	4	5	6	7	17	12:46
g14	800	4694	5	5	6	6	20	2:30
g19	800	4661	5	5	6	6	15	1:11
g23	2000	19990	24	24	29	32	1:59	2:50
g31	2000	19990	22	22	28	25	1:45	11:40
g33	2000	4000	15	15	20	30	2:47	3:27:25(†)
g38	2000	11779	17	18	21	19	1:28	9:02
g39	2000	11778	18	17	20	19	1:56	5:13
g44	1000	9990	10	10	13	17	46	1:04
g50	3000	6000	25	24	36	1:05	1:32	55
g52	1000	5916	7	7	8	8	42	3:14

Table 1: **Comparison of computing time.** Primal samples: S_{RP} random primal, S_{P} preswitching, S_{E} eigenspace; dual samples: S_{RD} random dual, S_{SD} dual sequence, and S_{ED} eigenspace after dual stop. The columns report the computing time in *hh*:*mm*:*ss*, *hh* hours, *mm* minutes, and *ss* seconds. *mm* reported when total seconds were more than 60, and similarly with *hh*. The run was performed with fixed steplength $\beta = 5$, minor iterations $n = 10$, major iteration horizon $M = 20$, uniform update on t, i.e. t values: $1/(M+1), \ldots, M/(M+1)$. (†) The excessive running time was due to the stopping criterion explained in section 3.4 on page 12.

heuristic primal sequence is not monotonically decreasing. Meaning that dual points closer to the optimum do not necessarily provide better heuristic primal points.

Table 3 shows a comparison with rank 2 and Goemans and Williamson technique using SeDuMi to solve the semidefinite formulation. MATLAB SeDuMi 1.03 [22] for SDP optimization tests were run on a machine with two 1GHz intel Pentium III (Coppermine) and 1Gb RAM. The numbers from rank 2 were extracted from the paper [5] from the table without the use of the random perturbation. There the results were obtained from a sample of size 1. In that paper, they report better results. However, we were not able to reproduce those results, since we did not know the information regarding the random perturbation parameters.

Figure 2: Plots of sequence of the values of dual points and their correspondent primal heuristic solution produced by Algorithm `pathfollowing` from `rudy` graphs g3, g14, and g22.



example	primal sampling			dual sampling			dual	G-W
name	S_{RP}	S_{P}	S_{E}	S_{RD}	S_{SD}	S_{ED}	bound	E(cut)
g3	99	99	99	99	99	11608	12084	10610
g6	99	100	100	100	100	2135	2656	
g13	100	100	100	100	99	568	647	
g14	99	99	99	99	99	3024	3192	2803
g19	98	98	98	99	102	868	1082	
g23	99	99	99	100	99	13234	14146	
g31	100	100	100	100	100	3170	4117	
g33	99	99	100	99	98	1342	1544	
g38	99	99	99	99	99	7512	8015	7037
g39	98	99	98	99	100	2258	2877	
g44	99	99	99	100	100	6601	7028	6170
g50	98	100	99	99	100	5830	5988	5257
g52	100	99	100	100	100	3779	4009	3520

Table 2: **Comparison of solution quality.** The first column shows the name of the problem. The following five columns are divided in primal samples (S_{RP} random primal, S_{P} preswitching, S_{E} eigenspace) and dual samples (S_{RD} random dual, S_{SD} dual sequence, and S_{ED} eigenspace after dual stop). The first five columns show the best case in percentage of the best value from the last sampling technique S_{ED}, in the last column whose result is reported in absolute value. The last two columns provide information about the dual bound and the expected cut using Goemans and Williamson heuristic. The expected cut was computed only for those graphs with non-negative edge weights. The run was performed with the same parameters as the previous table.

example	size		S_{RP}		SeDuMi		rank2	
name	n	m	ss	result	hh:mm:ss	result	$ss.ddd$	result
torusg3-8	512	1536	2	407	16:13	396	2.4†	403
g11	800	1600	4	550	53:12	530	.055	524
g12	800	1600	3	542	54:30	530	.063	512
g13	800	1600	4	570	52:15	556	.055	536
g14	800	4694	4	3006	55:41	2980	.09	3016
g15	800	4661	5	3002	1:10:29	2970	.09	3011
g20	800	4672	4	920	1:00:12	862	.113	901
g22	2000	19990	21	13193	14:46:44	12953	.363	13148
g24	2000	19990	25	13165	16:26:03	12963	.297	13195
g31	2000	19990	20	3193	N/A	N/A	.332	3146
g32	2000	4000	14	1346	14:25:52	1304	.176	1306
g34	2000	4000	14	1334	13:45:59	1288	.117	1276

Table 3: **Comparison with other methods.** Comparison of time and result among (i) random primal sampling S_{RP} with sample size = 10 and Algorithm `pathfollowing` with, steplength $\beta = 5$, fixed minor iterations $n = 10$ uniform update on t with major iteration horizon M = 20, t values: $1, (M-1)/M, (M-2)/M, ..., 1/M, 0$; (ii) SeDuMi 1.03 interior point plus Goemans and Williamson heuristic sample size 1000; and (iii) rank 2 heuristic, sample size equals to 1, (†) with the exception of the first example, torusg3-8, with sample size 100, where the average time was reported (total time was therefore multiplied by 100) and no perturbation used. The time is presented in ss seconds, hh:mm:ss hours:minutes:seconds, or $ss.ddd$ seconds.fraction expressed in decimal format. The results were rounded to the closest integer to ease the reading.

6 Conclusion

We presented a new heuristic for Max Cut, combining Lagrangian relaxation techniques with pathfollowing methods. The results we obtained are compared with previous results obtained by Semidefinite Programming using interior point algorithms, like SeDuMi, and special case algorithms like rank2. Apparently, the new method performs competitively with techniques previously mentioned.

The experienced running time is in the examples tested better than the one from interior point algorithm. However is not as good as the one from special purpose algorithm such rank 2 from Burer [5].

The idea presented here is generic and can be generalized, to all problems, where convex Lagrangian relaxation can be applied. The second author shows [19] that convex Lagrangian relaxations of general mixed integer quadratic programs can be obtained by solving an eigenvalue optimization problem. In order to apply Algorithm `pathfollowing` to this case the inner minimization and rounding has to be replaced by appropriate descent methods. This is currently under investigation.

Furthermore, there are several possibilities to accelerate the proposed method. First, decomposition techniques can be applied to solve the dual [19]. Second, Algorithm `pathfollowing` could be modified to trace the paths of all sample points simultaneously and delete candidates with high function values in an early stage of the pathfollowing. This approach is also well suited for parallelization.

It would be interesting to find out for which optimization problems the dual helps the algorithm to find better heuristic solutions. Similarly as discussed by Burer et al. [5], we found out that in the case of Max-Cut it might not be needed to solve the dual problem to improve the quality of heuristic solutions. It seems that the structure of the convex underestimators are not changed significantly by solving the dual. However, Max-Cut can be considered as a highly symmetric concave optimization problem with simple box-constraints. The situation could change for other optimization problems with more complicated constraints.

Acknowledgement. We would like to thank Prof. Kiwiel for making NOA 3.0 available, and Stefan Vigerske, who helped us with the `C++` coding and Linux scripting. The first author wants to thanks to Prof. Jane Dunphy for helping him in learning to write scientific work.

References

[1] J. E. Beasley, *Heuristic Algorithms for the Unconstrained Binary Quadratic Programming Problem*, tech. rep., The Management School, Imperial College, London SW7 2AZ, England, 1998. `http://mscmga.ms.ic.ac.uk/jeb/jeb.html`.

[2] S. J. Benson, Y. Ye, and X. Zhang, *Solving large-scale sparse semidefinite programs for combinatorial optimization*, SIAM J. Optim., 10(2) (2000), pp. 443–461.

[3] A. Berloni, P. Campadelli, and G. Grossi, *An approximation algorithm for the maximum cut problem and its experimental analysis*, Proceedings of "Algorithms and Experiments", (1998), pp. 137–143.

[4] D. Bertsimas and Y. Ye, *Semidefinite relaxations, multivariate normal distributions, and order statistics*, in Handbook of Combinatorial Optimization, D.-Z. Du and P. Pardalos, eds., Kluwer Academic Publishers, 1998, pp. 1–19.

[5] S. Burer, R. Monteiro, and Y. Zhang, *Rank-two Relaxation Heuristics for Max-Cut and Other Binary Quadratic Programs*, tech. rep., Department of Computational and Applied Mathematics Rice University, Houston, Texas 77005, November 2000. Technical Report TR00-33.

[6] S. Burer and R. D. C. Monteiro, *A Nonlinear Programming Algorithm for Solving Semidefinite Programs via Low-rank Factorization*, tech. rep., School of ISyE, Georgia Tech, Atlanta, March 2001.

[7] D. Dentcheva, J. Guddat, and J.-J. Rückmann, *Pathfollowing methods in nonlinear optimization III: multiplier embedding*, ZOR - Math. Methods of OR, 41 (1995), pp. 127–152.

[8] S. Feltenmark and K. C. Kiwiel, *Dual applications of proximal bundle methods including lagrangian relaxation of nonconvex problems*, SIAM J. Optim., 10(3) (2000), pp. 697–721.

[9] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, New York, 1979.

[10] M. X. Goemans and D. P. Williamson, *Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming*, J. ACM, 42 (1995), pp. 1115–1145.

[11] F. Gomes and D. Sorensen, `ARPACK++`*: a* `C++` *Implementation of* `ARPACK` *eigenvalue package*, 1997. `http://www.crpc.rice.edu/software/ARPACK/`.

[12] J. Guddat, F. Guerra, and D. Nowack, *On the role of the mangasarian-fromovitz constraints qualification for penalty-, exact penalty- and lagrange multiplier methods*, in Mathematical Programming with Data Perturbations, A. V. Fiacco, ed., Marcel Deckker, Inc., New York, 1998, pp. 159–183.

[13] J. Guddat, F. G. Vazquez, and H. T. Jongen, *Parametric Optimization: Singularities, Pathfollowing and Jumps*, John Wiley and Sons, 1990.

[14] C. Helmberg, *Semidefinite Programming for Combinatorial Optimization*, tech. rep., ZIB–Report 00–34, 2000.

[15] C. Helmberg and F. Rendl, *A spectral bundle method for semidefinite programming*, SIAM J. Optim., 10(3) (2000), pp. 673–695.

[16] J. B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms I and II*, Springer, Berlin, 1993.

[17] K. C. Kiwiel, *Proximity control in bundle methods for convex nondifferentiable minimization*, Math. Progr., 46 (1990), pp. 105–122.

[18] ——, *User's Guide for* `NOA 2.0/3.0`*: A* `FORTRAN` *Package for Convex Nondifferentiable Optimization*, Polish Academy of Science, System Research Institute, Warsaw, 1993/1994.

[19] I. Nowak, *Lagrangian Decomposition of Mixed-Integer All-Quadratic Programs*, tech. rep., HU–Berlin NR–2002–7, 2002.

[20] G. Pataki and S. H. Schmieta, *The DIMACS Library of Mixed Semidefinite Quadratic Linear Programs.* Instances available on the Internet. `http://dimacs.rutgers.edu/Challenges/Seventh/Instances/`.

[21] S. Schelstraete, W. Schepens, and H. Verschelde, *Energy minimization by smoothing techniques: a survey*, in From Classical to Quantum Methods, E. P. Balbuena and J. Seminario, eds., Elsevier, 1998, pp. 129–185.

[22] J. F. Sturm, *Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones*, Optimization Methods and Software, 11–12 (1999), pp. 625–653. Special issue on Interior Point Methods.