Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Journal of Computational and Applied Mathematics 182 (2005) 67–80
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

Probabilistic subproblem selection in branch-and-bound algorithms

Mirjam Dür[a,*], Volker Stix[b]

[a]*Department of Mathematics, Darmstadt University of Technology, D-64289 Darmstadt, Germany*
[b]*Department of Information Business, Vienna University of Economics and Business Administration, A-1090 Vienna, Austria*

Received 30 March 2004

Abstract

We investigate the branch-and-bound method for solving nonconvex optimization problems. Traditionally, much effort has been invested in improving the quality of the bounds and in the development of branching strategies, whereas little is known about good selection rules. After summarizing several known selection methods, we propose to introduce a probabilistic element into the selection process. We describe conditions which guarantee that a branch-and-bound algorithm using our probabilistic selection rule converges with probability 1. This new method is a generalization of the well-known best-bound selection rule. Furthermore, we relate the corresponding probability measure to the distribution of the optimal solution in the bounding interval. We also show how information on the quality of the upper and lower bounds influences the choice of the subset selection rule and conclude with numerical experiments on the Maximum Clique Problem which show that probabilistic selection can speed up an algorithm in many cases.

Keywords: Nonconvex programming; Subset selection in branch-and-bound algorithms; Convergence of branch-and-bound algorithms

* Corresponding author.
E-mail addresses: duer@mathematik.tu-darmstadt.de (M. Dür), Volker.Stix@wu-wien.ac.at (V. Stix).

0377-0427/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2004.10.019

1. Introduction and motivation

In this paper, we deal with the branch-and-bound method for maximizing a nonconcave function over a set of feasible points determined by the constraints. Formally, we consider the problem

$$\begin{aligned} &\max \quad f(x) \\ &\text{s.t. } x \in \mathscr{M}. \end{aligned} \tag{1}$$

It is common to assume the objective function f to be upper semicontinuous and the feasible set $\mathscr{M}$ to be compact, which ensures that the maximum exists.

This type of problem appears in numerous situations, and the branch-and-bound method is one possibility to solve it. Since it was first proposed in 1963 by Little et al. [12] this method has become increasingly popular and has been successfully applied to a wide range of applications. Just to mention a few examples: branch-and-bound has been used to solve discrete problems [2,3,7], continuous and stochastic problems [6,13–15], problems in economics and in technical applications [10,16,17].

To be able to refer to the particular steps of the branch-and-bound algorithm to solve problem (1), we recall its general framework (for a detailed introduction to the theory of branch-and-bound the reader is referred to Horst and Tuy [8]):

Algorithm.

Step 0: Compute a compact set $X \supset \mathscr{M}$ of simple structure (e.g., a simplex or hyperrectangle). Compute an upper bound $u(X) \geqslant \max_{x \in X \cap \mathscr{M}} f(x)$ for the solution of (1), as well as a lower bound $\ell(X) \leqslant \max_{x \in X \cap \mathscr{M}} f(x)$. Let $\mathscr{L}$ be an empty list, and set the iteration counter $k := 1$.

Step 1: Partition X into (a constant number of) v compact subsets $X_1, \ldots, X_v$, and add them to the list $\mathscr{L}$.

Step 2: Calculate upper bounds $u(X_i) \geqslant \max_{x \in X_i \cap \mathscr{M}} f(x)$ and lower bounds $\ell(X_i) \leqslant \max_{x \in X_i \cap \mathscr{M}} f(x)$ for all newly generated sets X_i.

Step 3: Update the current overall upper and lower bounds: Put

$$u_k := \max_{X \in \mathscr{L}} u(X) \quad \text{and} \quad \ell_k := \max_{X \in \mathscr{L}} \ell(X).$$

Step 4: Discard elements from the list $\mathscr{L}$ which cannot contain a global optimizer, i.e. discard all elements X with the property (i) $X \cap \mathscr{M} = \emptyset$, or (ii) $u(X) < \ell_k$.

Step 5: Select (according to some selection rule) a new $X \in \mathscr{L}$ which is to be subdivided in the next iteration, and remove it from $\mathscr{L}$.

Step 6: While stopping criteria are not fulfilled, increment $k := k + 1$, and go to step 1.

Our analysis will focus on step 5, the selection of the next subset, for which we discuss a new approach. Several possible selection rules are known: One popular procedure is depth search first. It is easy to implement and requires only little memory. In other implementations, problem dependent heuristics are used. Their main disadvantage is that convergence of the algorithm is often unsure.

A rule which ensures convergence of the branch-and-bound algorithm to the global maximum under mild conditions is the rule "select the set with the highest upper bound in each iteration" (see [8]). As we will often refer to this rule, we will call it "best-bound rule." We will call the set with the best upper bound in iteration k the dominating set of that iteration, and we will denote it by $\overline{X}_k$.

Recently, other selection criteria have been proposed: In the context of interval branch-and-bound methods, Csendes [4] suggested to choose that set for which a certain index computed from the available bounding information is maximal. He reports on good computational performance of his selection rule, cf. also Kreinovich and Csendes [11].

Stix [14] reports on performance improvements when a so-called target oriented branch-and-bound method is used. This means that not only the dominating set is selected for partitioning, but also all sets X whose $u(X)$ exceeds a certain value, the target value.

In the present paper, we propose to include a probabilistic element in the selection process. Our motivation is that we think that the traditional rule is too much focused on the dominating set. For example, in situations where there are several sets with upper bounds similar to the dominating set, there is no reason why the selection rule should discriminate those sets against the dominating one. Our idea is that randomization can somehow balance the selection process and speed up the algorithm runtime. Our numerical experiments on the Maximum Clique Problem show that this hope is justified in many cases.

To illustrate this idea, imagine that, at some iteration k, the list $\mathscr{L}$ consists of six elements $X_1, \dots, X_6$ whose upper and lower bounds can be represented as shown in Fig. 1.

Since $\ell_k \leqslant \max_{x\in\mathscr{M}} f(x) \leqslant u_k$, we have the feeling that some of the sets are "more likely" to contain the global solution than others. For example, in Fig. 1 the set X_3 seems "more likely" to contain the sought maximizer than X_5, whereas X_1 and X_2 are "almost even likely" to contain the solution. Therefore, we suggest to select the next set to be partitioned according to a probability distribution on $\mathscr{L}$ in such a way that sets with higher upper bounds get higher probabilities of being selected, but also sets with bad upper bounds are assigned a small, but nonzero probability of being chosen.

One possibility for such a probability distribution is to assign to each set a probability which is proportional to the length of the intersection of the two intervals $[\ell(X_i), u(X_i)]$ and $[\ell_k, u_k]$. This means that

$$\boldsymbol{P}_k(X_i \text{ is selected in iteration } k) \propto \frac{u(X_i) - \ell_k}{u_k - \ell_k}.$$

Obviously, this probability distribution depends on the list present in iteration k, a fact which is expressed by the notation $\boldsymbol{P}_k$. In order to ensure that $\boldsymbol{P}_k$ is a probability measure, it must be normed properly,

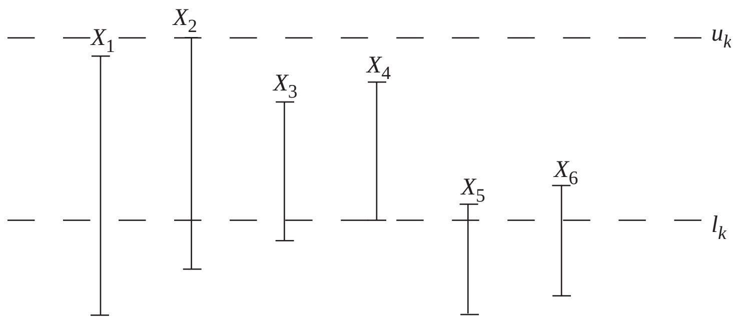


Fig. 1. A possible situation in iteration k, before step 5.

which leads to

$$\boldsymbol{P}_k(X_i \text{ is selected in iteration } k) = \frac{u(X_i) - \ell_k}{\sum_{X \in \mathscr{L}} (u(X) - \ell_k)}. \tag{2}$$

It is an important fact that we here assume all elements X of the list to fulfill $u(X) \geqslant \ell_k$, i.e., we assume that step 4 of the branch-and-bound algorithm has been carried out before introducing the probabilistic subset selection rule. This is equivalent to assigning the probability 0 to all sets X with $u(X) < \ell_k$. In later sections we will discuss other possible probability distributions on $\mathscr{L}$.

2. General setup and convergence results

It goes without saying that a probabilistic subset selection rule need not necessarily use the probability measure sketched above in (2), but can use other probability measures as well. One could think of a priori information on the quality of the bounds which could give rise to a certain probability measure. But also the traditional best-bound rule is a (degenerate) probabilistic rule: the corresponding probability measure assigns the probability 1 to the dominating set, and the probability 0 to all other sets. Other possible probability measures are discussed in Section 3.

In this section, we outline a general theory for probabilistic subset selection and give conditions under which the resulting branch-and-bound algorithms converge. An important assumption which we will presume to be satisfied throughout is that the random selections are independent in the sense that whether or not the dominating set is chosen in one iteration has no influence on whether or not it is selected in any other iteration. We first recall some important definitions.

Definition 1. A partitioning procedure is called exhaustive, if every nested sequence $\{X_i\}_{i \in \mathbb{N}}$ of partition sets eventually shrinks to a singleton $\{x^*\}$.

Most of the commonly used partitioning methods (e.g., bisection of simplices and rectangles) are exhaustive, cf. Horst and Tuy [8]. Furthermore, we need a concept which describes that, in the limit, lower and upper bounds of a sequence of partition sets coincide.

Definition 2. We say that a bounding procedure has the zero convergence property, if for every exhaustive sequence $\{X_i\}_{i \in \mathbb{N}}$ of subsets of X_0 we have

$$\lim_{i \to \infty} u(X_i) = \lim_{i \to \infty} \ell(X_i).$$

If a bounding procedure has this property, it follows immediately that both sequences of bounds converge to the maximum of f on the limit set $\bigcap_{i \in \mathbb{N}} X_i$: We have

$$\lim_{i \to \infty} u(X_i) \geqslant \lim_{i \to \infty} \max_{x \in X_i \cap \mathscr{M}} f(x) = \max_{x \in \bigcap\limits_{i \in \mathbb{N}} X_i \cap \mathscr{M}} f(x) \geqslant \lim_{i \to \infty} \ell(X_i),$$

where the inner equality follows from Lemma 2 in Dür [5]. The zero convergence property implies equality.

The following proposition gives an elementary convergence result. We understand a branch-and-bound procedure to be convergent if $\lim_{k \to \infty} u_k = \lim_{k \to \infty} \ell_k$.

Proposition 1. *Assume that in a branch-and-bound procedure*

(a) *a selection rule is used which ensures that the dominating set is chosen infinitely often,*
(b) *an exhaustive partitioning procedure is used, and*
(c) *the bounding procedure has the zero convergence property.*

Then the branch-and-bound procedure is convergent.

Proof. Denote by k_n the (infinite) subsequence of iterations where the dominating set is selected for subdivision, and by $\overline{X}_{k_n}$ the corresponding subsequence of dominating sets which, by definition, fulfill $u(\overline{X}_{k_n}) = u_{k_n}$. Since all $\overline{X}_{k_n}$ are nodes of the branch-and-bound tree which, at every depth k_n, has only a finite breadth, we conclude that there exists an infinite subsequence $\overline{X}_{k_{n_m}}$ of $\overline{X}_{k_n}$ which is nested. As the partitioning procedure is exhaustive, the diameters $d(\overline{X}_{k_{n_m}}) \to 0$. Therefore, using the zero convergence property, we have:

$$\lim_{k\to\infty} u_k = \lim_{m\to\infty} u_{k_{n_m}} = \lim_{m\to\infty} u(\overline{X}_{k_{n_m}}) = \lim_{m\to\infty} \ell(\overline{X}_{k_{n_m}}) \leqslant \lim_{m\to\infty} \ell_{k_{n_m}} = \lim_{k\to\infty} \ell_k.$$

Since $u_k \geqslant \ell_k$ for all k, the two limits must be equal, so the branch-and-bound procedure is convergent. □

The next theorem introduces the probabilistic subset selection into the theory. It gives a condition which guarantees convergence of the branch-and-bound procedure with probability 1.

Theorem 1. *Assume that in a branch-and-bound procedure*

(a) *a probabilistic selection rule is used which fulfills*

$$\sum_{k=0}^{\infty} \boldsymbol{P}_k(\overline{X}_k \textit{ is selected in iteration } k) = +\infty,$$

(b) *an exhaustive partitioning procedure is used, and*
(c) *the bounding procedure has the zero convergence property.*

Then the branch-and-bound procedure converges with probability 1.

Proof. Observe that in every iteration $k \in \mathbb{N}$ we are performing a Bernoulli experiment: Either we select the dominating set $\overline{X}_k$, or we do not, events which we can encode by 1 and 0, respectively. Hence, in every iteration k we have a sample space $\Omega_k = \Omega = \{0, 1\}$. The corresponding σ-field is the power set of Ω: $\mathscr{A}_k = \mathscr{A} = \{\emptyset, \{0\}, \{1\}, \Omega\}$. Thus, in every iteration we have the same measurable space $(\Omega, \mathscr{A})$. The probability measure $\boldsymbol{Q}_k$ describing the Bernoulli experiment, however, depends on the iteration. We define $\boldsymbol{Q}_k(\{1\})$ to be the probability that $\overline{X}_k$ is selected in iteration k:

$$\boldsymbol{Q}_k(\{1\}) = \boldsymbol{P}_k(\overline{X}_k \text{ is selected in iteration } k),$$

and $\boldsymbol{Q}_k(\{0\}) = 1 - \boldsymbol{Q}_k(\{1\})$. So we have defined a probability space $(\Omega, \mathscr{A}, \boldsymbol{Q}_k)$ for each iteration k.

Next, we want to describe the entire selection process, i.e., an infinite sequence of Bernoulli experiments. To this end, we need a common probability space, and the appropriate one is

$$\left(\prod_{k\in\mathbb{N}} \Omega, \bigotimes_{k\in\mathbb{N}} \mathscr{A}, \bigotimes_{k\in\mathbb{N}} \boldsymbol{Q}_k\right),$$

where $\prod_{k\in\mathbb{N}} \Omega$ denotes the infinite product $\Omega \times \Omega \times \dots$, (i.e., $\prod_{k\in\mathbb{N}} \Omega$ is the set of all sequences of elements out of $\{0,1\}$), $\bigotimes_{k\in\mathbb{N}} \mathscr{A}$ is the product σ-field, and $\bigotimes_{k\in\mathbb{N}} \boldsymbol{Q}_k$ is the product of the measures $\boldsymbol{Q}_k$. For details on these probabilistic concepts, the reader is referred to Bauer [1]. We use the abbreviation $\boldsymbol{Q} := \bigotimes_{k\in\mathbb{N}} \boldsymbol{Q}_k$.

Now let A_k be the event "$\overline{X}_k$ is selected in iteration k". Formally, $A_k \in \bigotimes_{k\in\mathbb{N}} \mathscr{A}$ is the event

$$A_k = \underbrace{\Omega \times \dots \times \Omega}_{k-1} \times \{1\} \times \Omega \times \cdots .$$

Its probability is $\boldsymbol{Q}(A_k) = \boldsymbol{Q}_k(\{1\})$. To see that the family of events $\{A_k\}_{k\in\mathbb{N}}$ is independent, take any nonempty finite subset $I = \{i_1, \dots, i_n\}$ of $\mathbb{N}$, and observe that

$$\boldsymbol{Q}(A_{i_1} \cap \dots \cap A_{i_n}) = \boldsymbol{Q}(A_{i_1}) \cdot \ldots \cdot \boldsymbol{Q}(A_{i_n}),$$

a basic property of the product measure.

Assumption (a) of the theorem translates to $\sum_{k=0}^{\infty} \boldsymbol{Q}(A_k) = +\infty$. Hence, we have an independent family $\{A_k\}_{k\in\mathbb{N}}$ of events which fulfill the preconditions of the Borel–Cantelli lemma (cf. again Bauer [1]). From this lemma, we conclude that with probability 1 infinitely many A_k occur, i.e., with probability 1 the dominating set is selected infinitely often. Combined with Proposition 1, this completes the proof of Theorem 1. □

Remark 1. The traditional best-bound rule trivially fulfills the conditions of Proposition 1 and Theorem 1. Thus our new probabilistic selection method is a generalization of the best-bound rule.

Remark 2. In some situations, the branch-and-bound tree is not an infinite, but a finite tree. This occurs, e.g., if at some depth of the tree the bounds $u(X)$ are known to be exact, or if in each iteration the problem dimension is reduced, such that for the subproblems corresponding to the leaves of the tree the dimension is 1. Moreover, the tree is finite in all practical situations where the problem is not solved exactly but to a prescribed tolerance $\varepsilon > 0$.

It is obvious that in the finite tree setting the described probabilistic subset selection results in a convergent, i.e., finite, algorithm.

Remark 3. Even if conditions (b) and (c) of Theorem 1 are fulfilled, it may occur that lower bounds $\ell_k = -\infty$, for example if lower bounds are computed using feasible points, but no feasible point can be found until iteration k. In such a situation, our randomized selection rule still results in a convergent algorithm, provided that the $\boldsymbol{P}_k$ fulfill condition (a) of Theorem 1.

One possibility to accomplish this is to use the best-bound-rule in all iterations k with $\ell_k = -\infty$, and to switch to a different measure once the lower bound becomes finite. Or one could use any probability measure $\boldsymbol{P}_k$ which assigns to the dominating set a higher probability than to every other set in the current list (cf. Corollary 1 below). Both ways will render the branch-and-bound procedure convergent.

Notice that under the assumption of the zero convergence property, if the algorithm generates an infinite nested sequence of partition sets, then there exists at least a subsequence of upper bounds converging to the optimal value of the underlying problem. Therefore, in the case where the problem is infeasible, the algorithm must terminate after finitely many iterations indicating this fact.

3. Possible realizations of the probability measure

3.1. A general concept

To see that the probability measure introduced in Section 1, which assigns probabilities according to (2), fulfills the key assumption (a) of Theorem 1, observe that this measure assigns to the dominating set the highest probability among all $X \in \mathscr{L}$. This means that this probability is not smaller than $1/|\mathscr{L}|$. Now in iteration k, the list contains at most kv elements (a consequence of step 1 of the branch-and-bound framework, where each selected set is partitioned into exactly v subsets), hence

$$\boldsymbol{P}_k(\overline{X}_k \text{ is selected in iteration } k) \geqslant \frac{1}{kv}.$$

Therefore,

$$\sum_{k=0}^{\infty} \boldsymbol{P}_k(\overline{X}_k \text{ is selected in iteration } k) \geqslant \frac{1}{v} \sum_{k=0}^{\infty} \frac{1}{k} = +\infty,$$

which is the decisive property in Theorem 1. More general, we have the following result:

Corollary 1. *Assume that in a branch-and-bound procedure*

(a) *a selection rule is used which ensures that in each iteration the dominating set is assigned the highest probability among all elements of the list,*
(b) *the number of elements in the list increases at most linearly in the iterations,*
(c) *an exhaustive partitioning procedure is used, and*
(d) *the bounding procedure has the zero convergence property.*

Then the branch-and-bound procedure is convergent with probability 1.

In the construction of the probability measure (2) we were implicitly assuming that the unknown optimal value $\max_{x \in \mathscr{M}} f(x)$ is distributed uniformly in the bounding interval. This is, of course, the natural thing to do if no additional information is available. But one could imagine that prior knowledge on the quality of the bounds is available which suggests to use a distribution different from the uniform one.

To be more precise, what we are doing here is (in a sort of Bayesian approach) to regard the unknown optimal value of (1) as a random variable Y whose distribution is described through a cumulative probability distribution function F. The distribution function F is viewed as a function telling us where in the bounding interval the unknown optimal value $\max_{x \in \mathscr{M}} f(x)$ is likely to be found and where it is not. Without loss of generality assume that Y is distributed in the interval [0,1].

Using the distribution function F, the probability measure $\boldsymbol{P}'_k$ must fulfill

$$\boldsymbol{P}'_k(X_i \text{ is selected in iteration } k) \propto F\left(\frac{u(X_i)-\ell_k}{u_k-\ell_k}\right).$$

Here the distribution function F has to be rescaled to the interval $[\ell_k, u_k]$. Norming to a probability measure gives

$$\boldsymbol{P}'_k(X_i \text{ is selected in iteration } k) = \frac{F((u(X_i)-\ell_k)/(u_k-\ell_k))}{\sum_{X\in\mathscr{L}} F((u(X)-\ell_k)/(u_k-\ell_k))}. \tag{3}$$

From the monotonicity of F we conclude that, compared to the probability measure (2), introducing a distribution function does not change the order of the subsets according to their probabilities of being selected in a given iteration k:

$$\boldsymbol{P}_k(X_i)\leqslant \boldsymbol{P}_k(X_j) \Longleftrightarrow u(X_i)\leqslant u(X_j) \Longleftrightarrow F\left(\frac{u(X_i)-\ell_k}{u_k-\ell_k}\right) \leqslant F\left(\frac{u(X_j)-\ell_k}{u_k-\ell_k}\right)$$
$$\Longleftrightarrow \boldsymbol{P}'_k(X_i)\leqslant \boldsymbol{P}'_k(X_j).$$

Therefore, the dominating subset remains the set with the highest probability among all elements of the list. As seen above, this property is sufficient for condition (a) of Theorem 1 to be fulfilled. This proves the following result:

Proposition 2. *Any choice of a distribution function F, combined with* (3), *results in a selection rule which ensures the algorithm to converge with probability* 1.

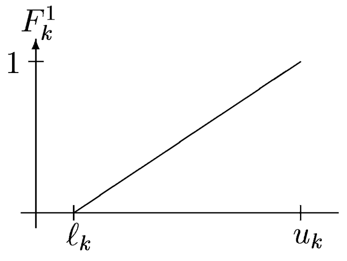
3.2. Examples

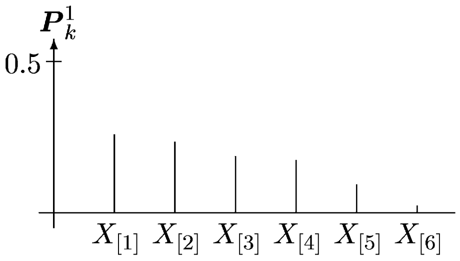
Next, we discuss different possible choices of distributions and demonstrate their effect on the probabilistic subset selection rule. To this end, take again the example of Fig. 1, but order the sets according to their upper bounds: Let $X_{[1]}$ denote the set with the highest upper bound, $X_{[6]}$ the set with the lowest.

As mentioned before, assuming F to be the uniform distribution function results in the probability measure introduced in (2): In this case, we have $F^1(x) = x$, whence

$$\boldsymbol{P}'_k(X_i \text{ is selected in iteration } k) = \frac{F^1((u(X_i)-\ell_k)/(u_k-\ell_k))}{\sum_{X\in\mathscr{L}} F^1((u(X)-\ell_k)/(u_k-\ell_k))} = \frac{u(X_i)-\ell_k}{\sum_{X\in\mathscr{L}}(u(X)-\ell_k)},$$

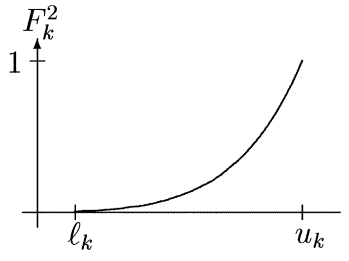
as asserted. The resulting probability distribution on $\mathscr{L}$ (along with the distribution function F^1_k which is the function F^1 rescaled to the interval $[\ell_k, u_k]$) can be seen in the following picture.

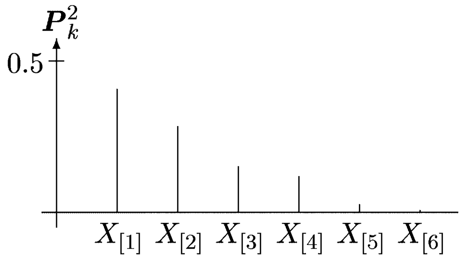




It may be reasonable to use other distributions, for example the distribution function F^2 shown below. This is reasonable if the upper bounds are supposed to be of higher quality than the lower bounds.

The resulting probabilities (calculated again for the example of Fig. 1) are sketched below. The difference to the uniform distribution is not too big, a fact which is not astonishing in view of the results of the next section. But note that the probability of the dominating set has increased compared to the uniform distribution.



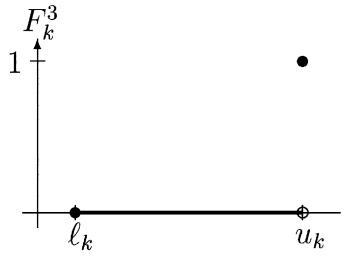


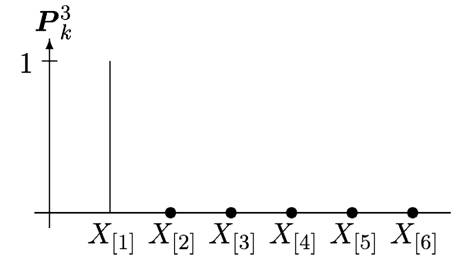
3.3. Deterministic versus random selection

Next, we investigate the two stochastically extreme distribution functions: Consider

$$F^3(x) = \begin{cases} 0 \ \ldots \ 0 \leqslant x < 1, \\ 1 \ \ldots \ x = 1. \end{cases}$$

F^3 corresponds to the situation where the global upper bound u_k is known to be sharp, i.e., it is known that $\max_{x \in \mathscr{M} \cap X} f(x) = u(X)$. Of course, this is only a hypothetical situation, because in this case, $\max_{x \in \mathscr{M}} f(x) = \max_{X \in \mathscr{L}} u(X) = u_k$, so the branch-and-bound algorithm would finish in the first iteration. Anyhow, it is clear that the dominating set should be preferred to all others. And this is exactly what the probability measure corresponding to F^3 defines: The resulting probabilities are 1 for the dominating set and 0 for all others, so this choice of distribution function yields the deterministic best-bound selection rule.





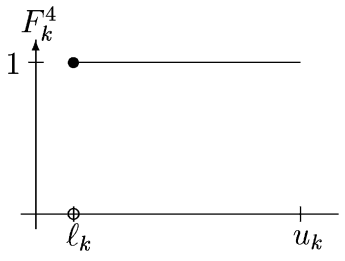
Now let us see what the other stochastically extreme distribution function leads to: Let

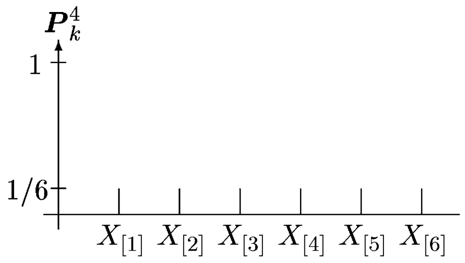
$$F^4(x) = 1 \quad \text{for} \quad 0 \leqslant x \leqslant 1.$$

F^4 corresponds to a setting where the lower bound ℓ_k is known to be sharp, which may occur if the optimal solution of the problem is known but the maximizing point is unknown, but also if good local solutions are available which provide good lower bounds. Since ℓ_k is a feasible objective value for all sets in the list, all of them should be assigned a positive probability. Indeed, here we get using (3)

$$\boldsymbol{P}_k^4(X_i \text{ is selected in iteration } k) = \frac{1}{\sum_{X\in\mathscr{L}} 1} = \frac{1}{|\mathscr{L}|},$$

so all sets from the list have the same probability of being chosen. This is somehow the purely random selection, whereas the stochastically smallest distribution function F^3 resulted in the purely deterministic choice.





We see that there is a range of selection rules, from purely deterministic to purely random, and these selection rules correspond to the two distribution functions F^3 and F^4. Every other distribution function F can be ranked on this scale. An appropriate way to measure the distance Δ of F to F^4 is

$$\Delta(F, F^4) = \int_0^1 (F^4(y) - F(y))\,\mathrm{d}y = \int_0^1 (1 - F(y))\,\mathrm{d}y = \boldsymbol{E}_F(Y),$$

where $\boldsymbol{E}_F(Y)$ denotes the expected value of the random variable Y (the unknown optimal value of the original problem (1), rescaled to the interval [0,1]) with respect to the distribution described by F. This quantity lies between 0 and 1. If the measure is near 0, this signifies that the selection procedure is highly random, a measure near 1 corresponds to a selection process which is very close to deterministic.

This means that if the expected value $\boldsymbol{E}_F(Y)$ is nearly 1 (= near to the upper bound), then it is recommendable to use a more deterministic selection procedure, if $\boldsymbol{E}_F(Y)$ is nearly 0 (= near to the lower bound), then a more random selection rule should be used.

4. Experimental results

We tested the implication of this new selection rule compared to the best-bound rule. The experiments were made on an NP-complete discrete problem, the Maximum Clique Problem (MCP), a well-known problem in graph theory. Its objective is to find the largest complete subgraph inside a given graph. As problem instances some graphs from the DIMACS challenge were chosen (see [9]). These graphs are known to be hard to solve with respect to the MCP.

The two compared algorithms have the same structure except for the selection step 5 (i.e., step 5 in the prototype algorithm of Section 1). On one hand we tested an algorithm (called “best-bound” below) which deterministically always selects the subproblem with the best upper bound. Our second algorithm selects the subproblem depending on an assumed distribution of the real maximum as introduced in this paper. The latter algorithm is called “random” below.

The following three figures (Figs. 2–4) illustrate three typical situations: All of them plot the number of iterations (i.e., sub-nodes visited) on the horizontal axis against the maximal clique-size found.

The best-bound algorithm is plotted using a dashed line whereas the random-algorithm is plotted with a solid line.

Fig. 2 shows the behavior of the two versions of the algorithm on the `san200_0.9_1` graph, a graph of dimension 200 (i.e., 200 vertices), density 0.9 (i.e., 90% of the possible edges are actually present), and a maximum clique size of 70. Here the random-algorithm is able to find larger cliques in earlier iterations than the best-bound algorithm all the time. This results in a better performance of this particular random-instance of 74%.

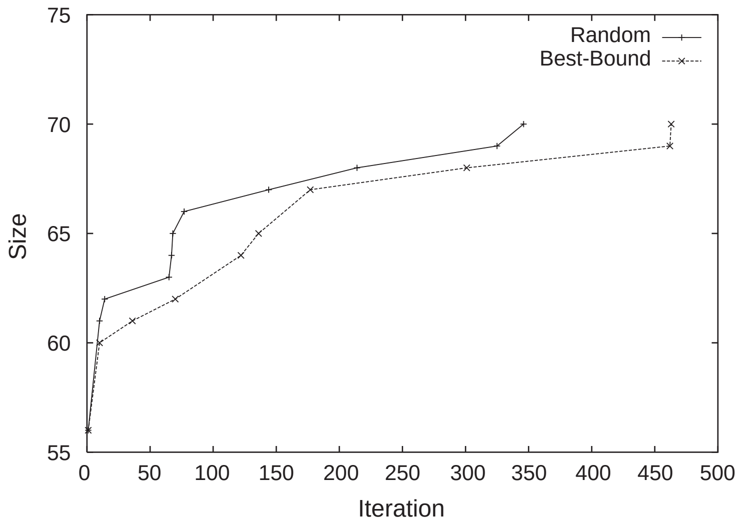


Fig. 2. Overall better performance of the random-algorithm.

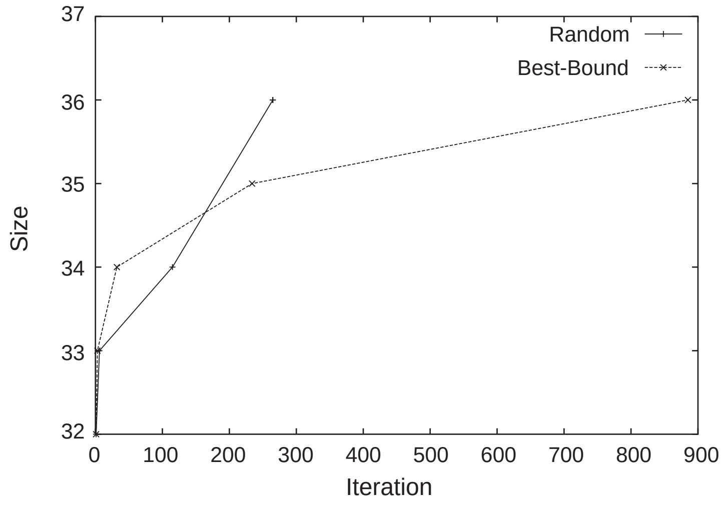


Fig. 3. Better performance of the random-algorithm.

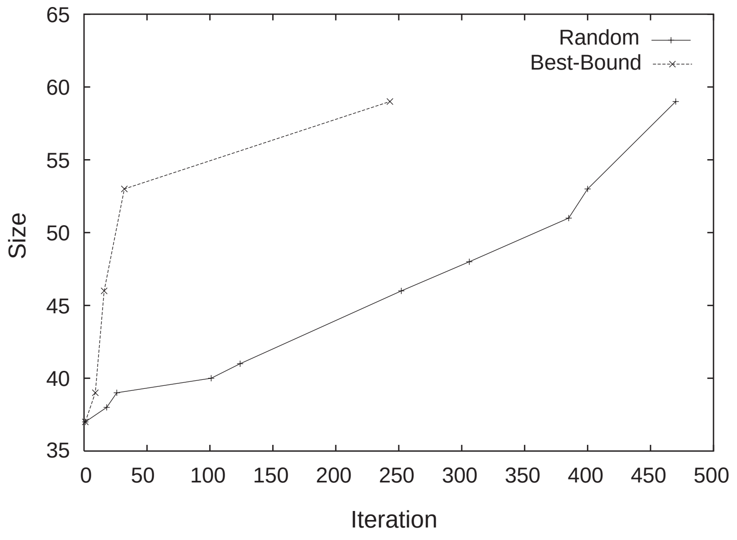


Fig. 4. Overall better performance of the best-bound algorithm.

Fig. 3 shows the results when testing `san200_0.9_3`, a different graph of dimension 200 and density 0.9. Here the maximum clique size is known to be 44. Both algorithms were terminated after 3000 iterations with the same maximal clique size of 36 found so far. It can be seen that although the best-bound algorithm was superior at the very beginning it was beaten by the random-algorithm. After 1000 iterations the best-bound algorithm had still around 58,000 subproblems in its list $\mathscr{L}$ (see Section 1) whereas the random-algorithm had only around 40,000. This phenomenon was observed in all of our experiments and will be discussed below.

Fig. 4 shows the results of a third graph of dimension 200 and density 0.9, the `san200_0.9_2` graph. Here the maximum clique size is 60. Both algorithms were stopped after 3000 iterations. At that stage, neither of them had found the maximum clique, but both of them had found a maximal clique of size 59. For this problem, the best-bound algorithm was superior compared to the random-algorithm. The same was observed after repeating the experiment with new random selections, whence we think that this is a structural property of that particular graph. After 1000 iterations the best-bound algorithm and random-algorithm had around 8700 and 6500 subproblems in their respective lists. We let the algorithms continue to run to check which one will first find the maximum clique, but without success after 3000 iterations.

The above plots illustrate some representative outcomes of our experiments. To give a more complete picture, we have summarized our experimental findings in Table 1.

The table lists graphs of dimension up to 400 for which both algorithms were unable to find the exact solution. Instead, both algorithms were stopped when they reached the same local solution so far. The time (measured in iterations) when the stopping happened is illustrated in Table 1. To reflect randomness, the random algorithm was restarted 20 times with different random seed.

The columns “min”, “avg” and “max” represent the minimum, average and maximum stopping-time for this algorithm, respectively. Column “best-bound” shows the stopping-time for the best-bound algorithm, and column “performance” gives the relative average time the random selection rule was faster compared

Table 1
Experimental results on DIMACS-graphs up to dimension 400

Name	Dimension	Random			Best-bound	Performance (%)
		Min.	Avg.	Max.		
brock200_1	200	**1497**	**1935.45**	**2103**	2802	69
brock400_1	400	**25**	**72.85**	**102**	105	69
brock400_2	400	**17**	**67.20**	107	90	71
brock400_3	400	**41**	163.45	259	99	165
brock400_4	400	**8**	**254.45**	560	330	77
MANN_a27	378	**9**	**16.25**	**30**	123	13
p_hat300-2	300	**2833**	4603.75	5079	3628	127
p_hat300-3	300	**53**	**145.10**	288	156	93
san200_0.9_2	200	**136**	253.90	348	243	104
san200_0.9_3	200	**92**	**853.20**	1732	885	96
san400_0.7_1	400	**23**	142.30	207	88	162
san400_0.7_2	400	**688**	**1046.60**	**1865**	2777	38
san400_0.7_3	400	**38**	833.70	1291	462	181
san400_0.9_1	400	**18**	**192.10**	511	272	71
sanr200_0.7	200	**12**	**77.25**	**132**	144	53
sanr200_0.9	200	**694**	**3468.05**	8613	3512	98
sanr400_0.7	400	**44**	**216.90**	372	257	84

to the best-bound rule (i.e., the ratio of the columns "avg" and "best-bound"). Instances where the random algorithm performed better than the best-bound algorithm are shown in bold.

It can be seen that in 12 out of the 17 testproblems the random selection rule performed (in average) better than the best-bound rule, in five cases it was even better with respect to its maximum stopping-time.

In five of our test problems the random selection rule performed worse (in average). In none of these five cases, however, it performed worse with respect to its minimum stopping-time.

An observation in all our experiments was that the size of the list $\mathscr{L}$ in the random-algorithm was always by a magnitude smaller than that of the best-bound algorithm. One explanation for that behavior is, that, for our test problems, larger problems tend to have a larger upper bound and tend to create more subproblems (which are nevertheless bounded by a fixed number). If the dominating set is always extracted next, it is more likely to create more subproblems than in the random-algorithm. A second explanation which is more general and not specific to a problem-class is as follows: by selecting randomly the probability of creating more infeasible subproblems is higher. These problems are removed from the list by step 4 of the general algorithm (see Section 1). Anyway, because of a smaller list $\mathscr{L}$, the random-algorithm consumes less memory and is therefore faster (e.g., in step 4).

We also found that the random-algorithm finds better lower bounds (local solutions) in earlier iterations and consumes less memory. For our problems an assumed density function similar to F_k^2 seemed to be more efficient than an equi-distribution implied by F_k^1 (see Section 3.2).

Therefore, our numerical experiments suggest that the probabilistic selection method has two main advantages: (i) better local optimizers and thus larger lower bounds are found in earlier iterations of the algorithm and (ii) the list $\mathscr{L}$ of candidate-sets is considerably smaller compared to the best-bound rule. These two features result in a speedup of the algorithm performance.

5. Conclusion

This article generalized the well-established best-bound selection rule using new probabilistic parameters. We showed convergence conditions for the probabilistic branch-and-bound algorithm and motivated different possible random distributions. Numerical tests performed on the Maximum Clique Problem showed that probabilistic selection often results in a speed-up of the algorithm.

Acknowledgements

The authors are indebted to the anonymous referees for valuable comments and suggestions, which have greatly improved the earlier version of this paper.

References

[1] H. Bauer, Probability Theory, de Gruyter, Berlin, 1996.
[2] R. Bausch, A multicriteria scheduling tool using a branch-and-bound algorithm, European J. Oper. Res. 61 (1992) 215–218.
[3] I.M. Bomze, V. Stix, Genetic engineering via negative fitness: evolutionary dynamics for global optimization, Ann. Oper. Res. 89 (1999) 297–318.
[4] T. Csendes, New subinterval selection criteria for interval global optimization, J. Global Optim. 19 (2001) 307–327.
[5] M. Dür, Dual bounding procedures lead to convergent branch-and-bound algorithms, Math. Programming 91 (2001) 117–125.
[6] M. Dür, R. Horst, N.V. Thoai, Solving sum-of-ratios fractional programs using efficient points, Optimization 49 (2001) 447–466.
[7] P. Hahn, T. Grant, N. Hall, A branch-and-bound algorithm for the quadratic assignment problem based on the Hungarian method, European J. Oper. Res. 108 (1998) 629–640.
[8] R. Horst, H. Tuy, Global Optimization: Deterministic Approaches, 3rd ed., Springer, Berlin, 1996.
[9] D.S. Johnson, M.A. Trick (Eds.), Cliques, coloring and satisfiability: Second dimacs implementation challenge, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 26, American Mathematical Society, Providence, RI, 1996.
[10] J. Karkazis, A branch-and-bound algorithm for the location of facilities causing atmospheric pollution, European J. Oper. Res. 58 (1992) 363–373.
[11] V. Kreinovich, T. Csendes, Theoretical justification of a heuristic subbox selection criterion for interval global optimization, Central European J. Oper. Res. 9 (2001) 255–265.
[12] J.D.C. Little, K.G. Murty, D.W. Sweeney, C. Karel, An algorithm for the traveling salesman problem, Oper. Res. 21 (1963) 972–989.
[13] V. Norkin, G. Pflug, A. Ruszczyński, A branch-and-bound method for stochastic global optimization, Math. Programming 83 (1998) 425–450.
[14] V. Stix, Target oriented branch & bound method for global optimization, J. Global Optim. 26 (2003) 261–277.
[15] V. Stix, Stochastic Branch & Bound applying Target Oriented Branch & Bound Method to Optimal Scenario Tree Reduction, Technical Report TR03/2002, Department of Information Business, Vienna University of Economics and Business Administration.
[16] M. Stojković, F. Soumis, An optimization model for the simultaneous operational flight and pilot scheduling problem, Management Sci. 47 (9) (2001) 1290–1305.
[17] F. Vanderbeck, A nested decomposition approach to a three-stage, two-dimensional cutting-stock problem, Management Sci. 47 (6) (2001) 864–879.