On a Particularity in Model-Based Search

Christian Blum[*], Michael Sampels
IRIDIA
Université Libre de Bruxelles
Brussels, Belgium
e-mail: {cblum,msampels}@ulb.ac.be
tel: (32) 2 650 3168, fax: (32) 2 650 2715

Mark Zlochin
Department of Computer Science
Technion – Israel Institute of Technology,
Haifa, Israel
e-mail: zmark@cs.technion.ac.il
tel: (972) 4 829 4948

Abstract

Giving positive feedback to good solutions is a common base technique in model-based search algorithms, such as Ant Colony Optimization, Estimation of Distribution Algorithms, or Neural Networks. In particular, the reinforcement of components of good solutions by positive feedback is known as a successful technique in tackling hard combinatorial optimization problems. We show by a simple model-based search algorithm for the node-weighted k-cardinality tree problem that this strategy doesn't guarantee steadily increasing performance of the algorithm in general. It is rather possible that for some "problem"-"probabilistic model" combinations the average performance of the system is decreasing and even the average probability of sampling good solutions is decreasing over time. The result is proven analytically and the consequences are studied in some empirical case studies.

1 Introduction

Model-based search (MBS) [10] algorithms are increasingly popular methods for solving combinatorial optimization problems. In MBS algorithms, such as Ant Colony Optimization (ACO) [2] or Estimation of Distribution Algorithms (EDAs)[1] [7, 8], candidate solutions are generated using a parametrized probabilistic model that is updated using the previously seen solutions in such a way that the search will concentrate in the regions of the search space containing high quality solutions. In particular, reinforcment of solution components depending on the solution quality is an important factor in the development of heuristics to tackle hard combinatorial optimization problems. It is assumed that good solutions don't occur sporadically, but consist of good solution components. To learn which components contribute to good solutions can help to assemble them to better solutions. In general, a model-based search approach attempts to solve an optimization problem by repeating the following two steps:

- Candidate solutions are constructed using some parametrized probabilistic model, that is, a parametrized probability distribution over the solution space.
- The candidate solutions are used to modify the model in a way that is deemed to bias future sampling toward low cost solutions.

Often it is implicitly assumed that the average performace of model-based algorithms is increasing over time. However, during empirical investigations of an Ant Colony Optimization Algorithm for the Group Shop scheduling problem[2] we observed that for some of the probabiltistic models chosen the performance of the system was decreasing over time. This triggered us to explore "problem"-"probabilistic model" combinations where the expected performance of a model-based search algorithm decreases over time. As a test case we chose the node-weighted k-cardinality tree problem, an NP-hard combinatorial optimization problem.

The paper is organized as follows. In Sec. 2 we briefly present the node-weighted k-cardinality tree problem. In Sec. 3 we outline a simple model-based search algorithm for the k-cardinality tree problem. Section 4 contains the analytical analysis of the system for a

[*] Corresponding author
[1] EDAs are covering several algorithms emerging from the field of Evolutionary Computation.

[2] Group Shop scheduling is a general formulation of scheduling problems covering Job Shop scheduling and Open Shop scheduling

small problem instance. Section 5 deals with empirical results and Sec. 6 finally offers conclusions and an outlook to the future.

2 The node-weighted k-cardinality tree problem

The k-cardinality tree problem is a combinatorial optimization problem that generalizes the well–known minimum weight spanning tree problem. It consists in finding in a node- or edge-weighted graph a subtree with exactly k edges, such that the sum of the weights is minimal. Due to various applications, such as oil-field leasing [6], facility layout [4], quorum-cast routing [1] and telecommunications [5], it has gained considerable interest in recent years. In this paper we deal with the k-cardinality tree problem in node-weighted graphs. The problem can be formally defined as follows. Let $G = (V, E)$ be a graph (where $|V| = n$ and $|E| = m$) with a weight function $w : V \to \mathbb{N}$ on the nodes. We denote the set of all k-cardinality trees in G by $\mathcal{T}_k$. Then the node-weighted problem (G, w, k) is to find a k-cardinality tree $T_k \in \mathcal{T}_k$ that minimizes

$$w(T_k) = \sum_{v \in V(T_k)} w(v). \tag{1}$$

The general problem is *NP*-hard, and in [9] *NP*-completeness results have been obtained for grid and split graphs.

3 A simple MBS algorithm for the k-cardinality tree problem

In this section we briefly outline a simple model-based search algorithm based on positive feedback for the k-cardinality tree problem. It is constructed in a straight-forward manner and it is representative for the class of model-based algorithms. The pseudo-code for this algorithm is shown in Alg. 1. In Alg. 1, T_k^i

Algorithm 1 A model-based algorithm for the node-weighted k-cardinality tree problem

```
input: A problem instance (G, w, k)
InitializeModelParameters(τ)
while termination conditions not met do
  for i = 1, ..., n_s do
    T_k^i ← ConstructSolution(τ)
  end for
  ApplyModelParameterUpdate(τ, T_k^1, ..., T_k^{n_s})
end while
output: A k-cardinality tree T_k^best
```

denotes the ith solution constructed in the current iteration, $n_s \geq 1$ is the total number of solutions constructed in every iteration, and $\tau = \{\tau_{v_1}, ..., \tau_{v_n}\}$ is a set of model parameters. After initialization of the model parameters, in every step of the algorithm n_s solutions are constructed using the current values of the model parameters. These solutions are then used to update the model parameters which are defined as follows: To every node $v \in V(G)$ we have associated a model parameter τ_v. The components of Alg. 1 are to be explained in more detail in the following.

InitializeModelParameters(τ): At the beginning of the algorithm, the model parameters τ_v are all initialized to the same small numerical value $c > 0$.

ConstructSolution(τ): In order to construct solutions to the problem we have to formalize how to use the model parameter values to construct solutions[3]. For constructing a solution, $k + 1$ construction steps are done as shown in Alg. 2. In every step of the construc-

Algorithm 2 Solution construction

```
V(T_0) ← ∅ and E(T_0) ← ∅
J_0 ← V(G)
Choose v* ∈ J_0 with probability p(v* | T_0){See eqn. (2)}
V(T_0) ← V(T_0) ∪ v*
for t = 1 to k do
  J_t ← {v ∈ V(G) \ V(T_{t-1}) | ∃ e[v_r, v] ∈ E(G) with v_r ∈ V(T_{t-1})}
  Choose v* ∈ J_t with probability p(v* | T_t)
  Find an edge e connecting v* with T_t
  E(T_t) ← E(T_{t-1}) ∪ {e}
  V(T_t) ← V(T_{t-1}) ∪ {v*}
end for
```

tion phase, we can only add a node v to the partial k-cardinality tree T_t ($t \in \{1, ..., k-1\}$) if there is an edge $e \in E(G)$ that connects one of the nodes in T_t with v. The probabilities $p(v_i \mid T_t)$ for all $v_i \in J_t$ are then defined in the following way.

$$p(v_i \mid T_t) = \begin{cases} \frac{\tau_{v_i}}{\sum_{v \in J_t} \tau_v} & \text{if} \quad v_i \in J_t \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where J_t is the set of nodes allowed to be added next to the partial k-cardinality tree T_t (see Alg. 2).

ApplyModelParameterUpdate(τ,T_k^1, . . . ,$T_k^{n_s}$): Once all solutions of an iteration are constructed, a rule updating the model parameters is applied. For the set of model parameters $\{\tau_{v_1}, ..., \tau_{v_n}\}$ this update rule is defined as:

$$\tau_{v_i} \leftarrow (1 - \rho) \cdot \tau_{v_i} + \tfrac{1}{n_s} \cdot \sum_{j=1}^{n_s} \Delta\tau_{v_i}^j \tag{3}$$

[3] We have to define a method of using the model to sample the search space.

where

$$\Delta\tau^j_{v_i} = \begin{cases} \frac{1}{w(T^j_k)} & \text{if } v_i \in T^j \\ 0 & \text{otherwise} \end{cases} . \quad (4)$$

In (4), T^j is the jth solution produced in the current iteration, $w(T^j)$ is the quality of solution T^j and $0 < \rho < 1$ is a parameter supporting the intensification of the search (a similar parameter in ACO algorithms is called evaporation rate). This update rule is similar to update rules used in Ant Colony Optimization and some other population-based methods from the field of Evolutionary Computation. It should lead to an increase of model parameter values associated with solution components which have been found in better quality solutions compared with other solution components, hence increasing the expected performanc of the algorithm over time. In the following section we will prove that this expectation is not always met in practice.

4 A counterexample: Decreasing average performance of a MBS algorithm over time

In this section we choose a small problem instance of the k-cardinality tree problem and we show that the expected performance of Alg. 1 is decreasing. The problem instance under consideration is shown in Fig. 1. It shows an undirected graph $G = (V, E)$ formally defined as follows.

$$V(G) = \{v_1, v_2, v_3, v_4\} \quad (5)$$
$$E(G) = \{e_{v_1,v_2}, e_{v_2,v_3}, e_{v_3,v_4}\} \quad (6)$$
$$w(v_i) = w_i > 0, \ \text{for } i = 1, ..., 4 \quad (7)$$

For the weights we choose

$$w_2, w_3 > w_1, w_4, \quad (8)$$

and for the sake of simplicity we choose

$$w_1 = w_4 \text{and } w_2 = w_3 \ . \quad (9)$$

In graph G we want to solve the 1-cardinality tree problem with the model-based search algorithm outlined in the last section. In G we can find 3 different 1-cardinality trees:

$$\begin{array}{lll} T^a & \text{with} & V(T^a) = \{v_1, v_2\}, E(T^a) = \{e_{v_1,v_2}\} \\ & \Rightarrow & w(T^a) = w_1 + w_2 \\ T^b & \text{with} & V(T^b) = \{v_2, v_3\}, E(T^b) = \{e_{v_2,v_3}\} \\ & \Rightarrow & w(T^b) = w_2 + w_3 \\ T^c & \text{with} & V(T^c) = \{v_3, v_4\}, E(T^c) = \{e_{v_3,v_4}\} \\ & \Rightarrow & w(T^c) = w_3 + w_4 \end{array}$$

Because of (8) and (9) it holds that

$$w(T^a) = w(T^c) < w(T^b). \quad (10)$$

Therefore T^a and T^c are both optimal solutions of this problem instance and T^b is the worst solution. With model parameter values $\tau_{v_1}(t), ..., \tau_{v_4}(t)$ at discrete time steps $t = 0, 1, ...$, the probabilities $p_a(t)$, $p_b(t)$ and $p_c(t)$ to construct solutions T^a, T^b and T^c are the following:

$$p_a(t) = \frac{\tau_{v_1}(t)}{\sum_{i=1}^4 \tau_{v_i}(t)} + \frac{\tau_{v_2}(t)}{\sum_{i=1}^4 \tau_{v_i}(t)} \cdot \frac{\tau_{v_1}(t)}{\tau_{v_1}(t)+\tau_{v_3}(t)} \quad (11)$$

$$p_b(t) = \frac{\tau_{v_2}(t)}{\sum_{i=1}^4 \tau_{v_i}(t)} \cdot \frac{\tau_{v_3}(t)}{\tau_{v_1}(t)+\tau_{v_3}(t)} + \frac{\tau_{v_3}(t)}{\sum_{i=1}^4 \tau_{v_i}(t)} \cdot \frac{\tau_{v_2}(t)}{\tau_{v_2}(t)+\tau_{v_4}(t)} \quad (12)$$

$$p_c(t) = \frac{\tau_{v_4}(t)}{\sum_{i=1}^4 \tau_{v_i}(t)} + \frac{\tau_{v_3}(t)}{\sum_{i=1}^4 \tau_{v_i}(t)} \cdot \frac{\tau_{v_4}(t)}{\tau_{v_4}(t)+\tau_{v_2}(t)} \quad (13)$$

We use the notation p_{v_i} for the probability of node v_i to be found in a solution constructed. These probabilities are obviously the following ones.

$$p_{v_1}(t) = p_a(t) \quad (14)$$
$$p_{v_2}(t) = p_a(t) + p_b(t) \quad (15)$$
$$p_{v_3}(t) = p_b(t) + p_c(t) \quad (16)$$
$$p_{v_4}(t) = p_c(t) \quad (17)$$

Let us now examine the evolution of the model parameter values over time.

Evolution of τ_{v_1} over time: After every construction step there are two possibilities. Either v_1 is a part of the constructed solution, or it is not. Then the expected value of τ_{v_1} at time $t + 1$ is the following.

$$E(\tau_{v_1}(t+1)) = \left((1-\rho) \cdot \tau_{v_1}(t) + \frac{1}{w_1+w_2}\right) \cdot p_{v_1}(t)$$

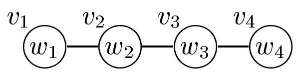

Figure 1: The problem instance consists of four nodes v_1, v_2, v_3 and v_4, connected by three edges. The node weights of the nodes v_i are indicated by w_i.

$$\begin{aligned}
&+ \quad (1-\rho)\cdot\tau_{v_1}(t)\cdot(1-p_{v_1}(t))\\
&= \quad \tau_{v_1}(t)p_{v_1}(t) - \rho\tau_{v_1}(t)p_{v_1}(t)\\
&+ \quad \tfrac{1}{w_1+w_2}p_{v_1}(t) + \tau_{v_1}(t) - \tau_{v_1}(t)p_{v_1}(t)\\
&- \quad \rho\tau_{v_1}(t) + \rho\tau_{v_1}(t)p_{v_1}(t)\\
&= \quad \tau_{v_1}(t) + \left(\tfrac{1}{w_1+w_2}\cdot p_{v_1}(t) - \rho\tau_{v_1}(t)\right)
\end{aligned}$$

As $p_{v_1}(t) = p_a(t)$ we get

$$E(\tau_{v_1}(t+1)) \;=\; \tau_{v_1}(t) + \left(\tfrac{1}{w_1+w_2}\cdot p_a(t) - \rho\tau_{v_1}(t)\right) \tag{18}$$

Evolution of τ_{v_2} over time: After every construction step there is – as above for τ_{v_1} – the possibility that v_2 is a part of the solution constructed. But there are also two different solutions v_2 can be part of. In the following p_{a+b} will stand for p_a+p_b, w_{1+2} will stand for w_1+w_2, and w_{2+3} will stand for w_2+w_3.

$$\begin{gathered}
E(\tau_{v_2}(t+1)) \;=\\
\left((1-\rho)\cdot\tau_{v_2}(t) + \tfrac{p_a(t)}{p_{a+b}(t)}\tfrac{1}{w_{1+2}} + \tfrac{p_b(t)}{p_{a+b}(t)}\tfrac{1}{w_{2+3}}\right)\cdot p_{v_2}(t)\\
+(1-\rho)\cdot\tau_{v_2}(t)\cdot(1-p_{v_2}(t))
\end{gathered}$$

As $p_{v_2} = p_{a+b}$ we get

$$\begin{aligned}
E(\tau_{v_2}(t+1)) &= (1-\rho)\cdot\tau_{v_2}(t)\cdot p_{a+b}(t) + \tfrac{p_a(t)}{w_{1+2}}\\
&+ \tfrac{p_b(t)}{w_{2+3}} + (1-\rho)\,\tau_{v_2}(t)\,(1-p_{a+b}(t))\\
&= \tau_{v_2}(t)p_{a+b}(t) - \rho\tau_{v_2}(t)p_{a+b}(t)\\
&+ \tfrac{p_a(t)}{w_{1+2}} + \tfrac{p_b(t)}{w_{2+3}} + \tau_{v_2}(t)\\
&- \tau_{v_2}(t)p_{a+b}(t) - \rho\tau_{v_2}(t)\\
&+ \rho\tau_{v_2}(t)p_{a+b}(t)\ .
\end{aligned}$$

Therefore we get

$$E(\tau_{v_2}(t+1)) \;=\; \tau_{v_2}(t) + \left(\tfrac{p_a(t)}{w_{1+2}} + \tfrac{p_b(t)}{w_{2+3}} - \rho\tau_{v_2}(t)\right)\ . \tag{19}$$

For the **evolution of τ_{v_3} and τ_{v_4}** the computations are the same as for τ_{v_2} and τ_{v_1} respectively. Consequently

$$E(\tau_{v_3}(t+1)) \;=\; \tau_{v_3}(t) + \left(\tfrac{p_c(t)}{w_{3+4}} + \tfrac{p_b(t)}{w_{2+3}} - \rho\tau_{v_3}(t)\right) \tag{20}$$

and

$$E(\tau_{v_4}(t+1)) \;=\; \tau_{v_4}(t) + \left(\tfrac{1}{w_{3+4}}\cdot p_c(t) - \rho\tau_{v_4}(t)\right)\ . \tag{21}$$

It is common practice in Genetic Algorithm research to analyse the bahaviour of the algorithm with infinite population size. Therefore, in the following we consider the limit case of $n_s\to\infty$. In this case the law of large number says that the actual value of $\tau_{v_i}(t)$ converges in probability to the expected value. Therefore we use $\tau_{v_i}(t)$ instead of $E(\tau_{v_i}(t))$ in the following. The algorithm at time $t=0$ starts with $\tau_{v_1}(0) = \tau_{v_2}(0) = \tau_{v_3}(0) = \tau_{v_4}(0) = c > 0$. With (9), (11) and (13) it follows that $p_a(0) = p_c(0)$. With (18) and (21) it follows that $\tau_{v_1}(1) = \tau_{v_4}(1)$ and with (19) and (20) it follows that $\tau_{v_2}(1) = \tau_{v_3}(1)$. In turn this implies with (9) that $p_a(1) = p_c(1)$. By induction it follows that for $t\ge 0$

$$\tau_{v_1}(t) \;=\; \tau_{v_4}(t) \quad\text{and}\quad \tau_{v_2}(t) = \tau_{v_3}(t) \tag{22}$$

$$p_a(t) \;=\; p_c(t)\ . \tag{23}$$

This means that the evolution of $\tau_{v_1}(t)$ is equal to the evolution of $\tau_{v_4}(t)$, the evolution of $\tau_{v_2}(t)$ is equal to the evolution of $\tau_{v_3}(t)$ and the probability to produce tree T^a is equal to the probability to produce tree T^c. This allows us to simplify equations (11), (12) and (13) expressing everything in terms of $\tau_{v_1}(t)$ and $\tau_{v_2}(t)$. Substituting $\tau_{v_3}(t)$ by $\tau_{v_2}(t)$ and $\tau_{v_4}(t)$ by $\tau_{v_1}(t)$ in equations (11), (12) and (13) results in

$$p_a(t) \;=\; p_c(t) \;=\; \tfrac{\tau_{v_1}(t)}{2(\tau_{v_1}(t)+\tau_{v_2}(t))}\cdot\left(1+\tfrac{\tau_{v_2}(t)}{\tau_{v_1}(t)+\tau_{v_2}(t)}\right) \tag{24}$$

$$p_b(t) \;=\; \tfrac{\tau_{v_2}(t)}{\tau_{v_1}(t)+\tau_{v_2}(t)}\cdot\tfrac{\tau_{v_2}(t)}{\tau_{v_1}(t)+\tau_{v_2}(t)}\ . \tag{25}$$

Theorem 1 *In the settings described above the following holds. The probability $p_b(t)$ to produce T^b (the worst 1-cardinality tree to be found in graph G as defined in equations (5)–(9)) is increasing from the first step of Alg. 1 onward as long as*

$$\tfrac{1}{w_{2+3}}p_b(t)\cdot\tau_{v_1} > \tfrac{1}{w_{1+2}}\cdot p_a(t)\,(\tau_{v_2}-\tau_{v_1})\ . \tag{26}$$

Proof: In the following we will write $\tau_{v_1+v_2}(t)$ for $\tau_{v_1}(t)+\tau_{v_2}(t)$. Then

$$\begin{gathered}
p_b(t+1) > p_b(t)\\
\Leftrightarrow\\
\left(\frac{\tau_{v_2}(t)+\left(\frac{p_a(t)}{w_{1+2}}+\frac{p_b(t)}{w_{2+3}}-\rho\tau_{v_2}(t)\right)}{\tau_{v_1+v_2}(t)+\left(\frac{2p_a(t)}{w_{1+2}}+\frac{p_b(t)}{w_{2+3}}-\rho(\tau_{v_1+v_2}(t))\right)}\right)^2\\
>\\
\left(\frac{\tau_{v_2}(t)}{\tau_{v_1+v_2}(t)}\right)^2\\
\Leftrightarrow \text{(taking the square root)}\\
\left(\tau_{v_2}(t)+\left(\tfrac{p_a(t)}{w_{1+2}}+\tfrac{p_b(t)}{w_{2+3}}-\rho\tau_{v_2}(t)\right)\right)\cdot(\tau_{v_1+v_2}(t))\\
>\\
\tau_{v_2}(t)\cdot\left(\tau_{v_1+v_2}(t)+\left(\tfrac{2p_a(t)}{w_{1+2}}+\tfrac{p_b(t)}{w_{2+3}}-\rho(\tau_{v_1+v_2}(t))\right)\right)\\
\Leftrightarrow\\
\tau_{v_2}(t)\tau_{v_1}(t)+(\tau_{v_2}(t))^2+\tfrac{p_a(t)}{w_{1+2}}\tau_{v_1}(t)+\tfrac{p_a(t)}{w_{1+2}}\tau_{v_2}(t)\\
+\tfrac{p_b(t)}{w_{2+3}}\tau_{v_1}(t)+\tfrac{p_b(t)}{w_{2+3}}\tau_{v_2}(t)-\rho\tau_{v_2}(t)\tau_{v_1}(t)-\rho\,(\tau_{v_2}(t))^2\\
>\\
\tau_{v_2}(t)\tau_{v_1}(t)+(\tau_{v_2}(t))^2+\tfrac{2p_a(t)}{w_{1+2}}\tau_{v_2}(t)+\tfrac{p_b(t)}{w_{2+3}}\tau_{v_2}(t)\\
-\rho\tau_{v_2}(t)\tau_{v_1}(t)-\rho\,(\tau_{v_2}(t))^2\\
\Leftrightarrow\\
\tfrac{1}{w_{2+3}}p_b(t)\cdot\tau_{v_1} > \tfrac{1}{w_{1+2}}\cdot p_a(t)\,(\tau_{v_2}-\tau_{v_1})\ .
\end{gathered}$$

As in the first iteration of Alg. 1 all model parameter values are equal and greater than 0, it holds that $p_b(1) > p_b(0)$. **qed**

Remark 1 *With (22) it is now also clear that in the case $p_b(t+1) > p_b(t)$ it holds that*

$$\begin{aligned} p_a(t+1) &< p_a(t), & (27)\\ p_c(t+1) &< p_c(t), & (28)\\ p_a(t+1) - p_a(t) &= -\tfrac{1}{2}\left(p_b(t+1) - p_b(t)\right), & (29)\\ p_c(t+1) - p_c(t) &= -\tfrac{1}{2}\left(p_b(t+1) - p_b(t)\right) & (30) \end{aligned}$$

If we now measure the performance $\mathcal{P}(t)$ of Alg. 1 as the expected average quality of a solution produced at any time $t \geq 0$ as

$$\mathcal{P}(t) = w_{1+2} \cdot p_a(t) + w_{2+3} \cdot p_b(t) + w_{3+4} \cdot p_c(t), \quad (31)$$

then it is clear that for $p_b(t+1) > p_b(t)$ the expected performance of the system is decreasing from time t to time $t+1$. Using Theorem 1 it follows that the expected performance of Alg. 1 is decreasing from $t = 0$ to an unknown time $t_{stop} > 0$ which may be finite or infinite. To summarize, we have shown that it is possible to find circumstances where the expected performance – as formalized in (31) – of Alg. 1 is decreasing. In the next section we will confirm the outcomes outlined in this section empirically.

5 Analytical curves and empirical confirmation

In this section we present analytical and empirical results for Alg. 1 on two different weight settings for the k-cardinality tree problem instance defined in equations (5)–(9). The first setting of the weights is

$$w_1 = w_4 = 1.0 \text{ and } w_2 = w_3 = 2.0 \quad (32)$$

further on referred to as problem instance 1. The second setting is

$$w_1 = w_4 = 1.0 \text{ and } w_2 = w_3 = 100.0 \quad (33)$$

further on referred to as problem instance 2. We used the formulas derived in the last section[4] to produce the analytical curves showing (i) the performance of the system as defined in (31), and (ii) the evolution of the four model parameter values over time. We also run Alg. 1 with 10000 solution constructions per iteration for four different values for parameter $\rho \in \{0.5, 0.1, 0.05, 0.01\}$ on both problem instances to see the empirical behavior of the system. The results are shown in Fig. 3–5. In Fig. 3 and Fig. 4, we observe

[4]Formulas (18)–(21) for the evolution of model parameter values, and formulas (11)–(13) for the evolution of the probabilities to produce the different solutions.

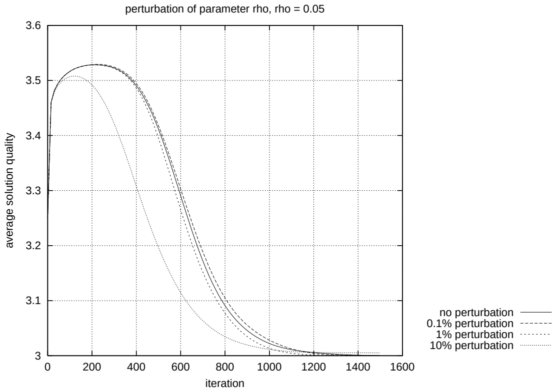

Figure 2: This graph shows four different empirically obtained performance curves for problem instance 1, $\rho = 0.05$ (averaged over 100 experiments). The curve labeled "no perturbation" is the curve for the system starting with model parameter values $\tau_{v_1} = \tau_{v_2} = \tau_{v_3} = \tau_{v_4} = c$, where c is the starting value for the model parameters. The other curves show the performance of the system when there is a random perturbation on the starting model parameters. 0.1% perturbation for example means that for every model parameter value we were tossing a coin: this means that to equal probability we either added $\frac{1}{1000} \cdot c$ to τ_{v_i} or we subtracted $\frac{1}{1000} \cdot c$ from τ_{v_i}. We did that for each of the four model parameter values independently.

that the performance of the system in the analytical and also in the empirical case decreases quite drastically in the first couple of hundred iterations. Then, the analytic curves seem to converge (with still decreasing performance) to a fixed point of the system (not being one of the solutions). This fixed point is implicitly defined by equation (26)[5]. On the other hand, the empirically obtained curves show even a slightly higher decrease in performance at the beginning, but then near the equilibrium the sensitivity to sampling errors seems to increase rapidly. As a result, the system converges to one of the two optimal solutions (T_a or T_b). For problem instance 2, the number of iterations needed for both, the analytical and the empirical curves, to reach there limits is lower than for problem instance 1. Qualitatively the results are the same for the two different weight settings. This is in accordance with the results of the last section, which are not dependent on the relative difference between weights $w_1 = w_4$ and $w_2 = w_3$.

[5]This claim is supported by the fact that if we take the analytically obtained convergence values (exact up to four decimal digits) for the four model parameter values and substitute them in equations (24), (25) and (26) we get equality in (26) for up to decimal digits.

We also compared the evolution of the four model parameter values over time, analytically as well as empirically. The graphs are shown in Fig. 5. In the analytic case, the evolution of model parameter values τ_{v_1} and τ_{v_4} (resp. τ_{v_2} and τ_{v_3}) is the same whereas in the empirical case the evolution is the same at the beginning, until the system nearly reaches the equilibrium and then, due to the sampling error, begins to drift toward one of the two optimal solutions.

The last set of experiments we performed was to investigate the influence of perturbations of the starting model parameter values. To perturb the model parameter values (initially set to a constant c), a value $\frac{1}{x} \cdot c$ was either added or substracted to equal probability. For x we considered values 10, 100 and 1000. The results are shown in Fig. 2. With $x = 1000$ (just a slight perturbation) the convergence of the system seems to be slightly slowed down, but at the end it is reaching the state of convergence slightly faster than the system without perturbation. With the choice of $x = 100$, we notice a bigger advantage in convergence speed in all phases. The choice of $x = 10$ (which corresponds to a strong perturbation) shows a much higher convergence speed until about 1000 iterations at which point the speed of convergence gets really slow and the system does not even converge before the 1500 iteration limit. These results suggest that a slight perturbation of the initial model parameter values is useful (but it must neither be too low nor too high).

6 Conclusions and outlook to the future

In this paper we showed that – unlike what is usually expected – the expected performance of model-based search algorithms using positive feedback can decrease over time in certain settings. We want to make clear at this point, that the results presented in this paper have not uncovered a general weak point of learning systems based on positive feedback as such. We believe that it is rather in the nature of algorithms such as Ant System [3] and related algorithms such as Population Based Incremental Learning (PBIL) and Estimation of Distribution algorithms (EDAs) that such phenomena can occur when applied to a certain kind of problem or problem instances with a certain kind of structure. In the case of the problem and problem instance chosen in this paper we have a structure of two equally good solutions competing in the system and a bad solution taking profit from that. We also point out that the update rule is a crucial component of a model-based search algorithm. If we chose a different update rule in the example presented – for instance a rule only using the best solution found since the start of the algorithm for updating the model parameter values – the algorithm wouldn't show a decreasing performance. In the future we intend to investigate into the interactions between parameter model and model parameter update rule in order to improve the understanding about phenomena related to the one presented.

Acknowledgments: This work was supported by the "Metaheuristics Network", a Research Training Network funded by the Improving Human Potential program of the CEC, grant HPRN-CT-1999-00106. The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication.

References

[1] S.Y. Cheung and A. Kumar. Efficient quorumcast routing algorithms. In *Proceedings of INFOCOM'94*, Los Alamitos, USA, 1994. IEEE Society Press.

[2] M. Dorigo, G. Di Caro, and L. M. Gambardella. Ant algorithms for discrete optimization. *Artificial Life*, 5(2):137–172, 1999.

[3] M. Dorigo, V. Maniezzo, and A. Colorni. Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics–Part B*, 26(1):29–41, 1996.

[4] L.R. Foulds and H.W. Hamacher. A new integer programming approach to (restricted) facilities layout problems allowing flexible facility shapes. Technical Report 1992-3, University of Waikato, Department of Management Science, 1992.

[5] N. Garg and D. Hochbaum. An $O(\log k)$ approximation algorithm for the k minimum spanning tree problem in the plane. *Algorithmica*, 18:111–121, 1997.

[6] H.W. Hamacher and K. Joernsten. Optimal relinquishment according to the Norwegian petrol law: A combinatorial optimization approach. Technical Report No. 7/93, Norwegian School of Economics and Business Administration, Bergen, Norway, 1993.

[7] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the 4th Conference on Parallel Problem Solving from Nature – PPSN IV*, volume 1411 of *Lecture Notes in Computer Science*, pages 178–187, Berlin, 1996. Springer.

[8] M. Pelikan, D.E. Goldberg, and F. Lobo. A survey of optimization by building and using probabilistic models. Technical Report No. 99018, IlliGAL, University of Illinois, 1999.

[9] G.J. Woeginger. Computing maximum valued regions. *Acta Cybernetica*, 4(10):303–315, 1992.

[10] M. Zlochin, M. Birattari, N. Meuleau, and M. Dorigo. Model-based search for combinatorial optimization. Technical Report TR/IRIDIA/2001-15, IRIDIA, Université Libre de Bruxelles, Belgium, 2001. Submitted to Annals of Operations Research.

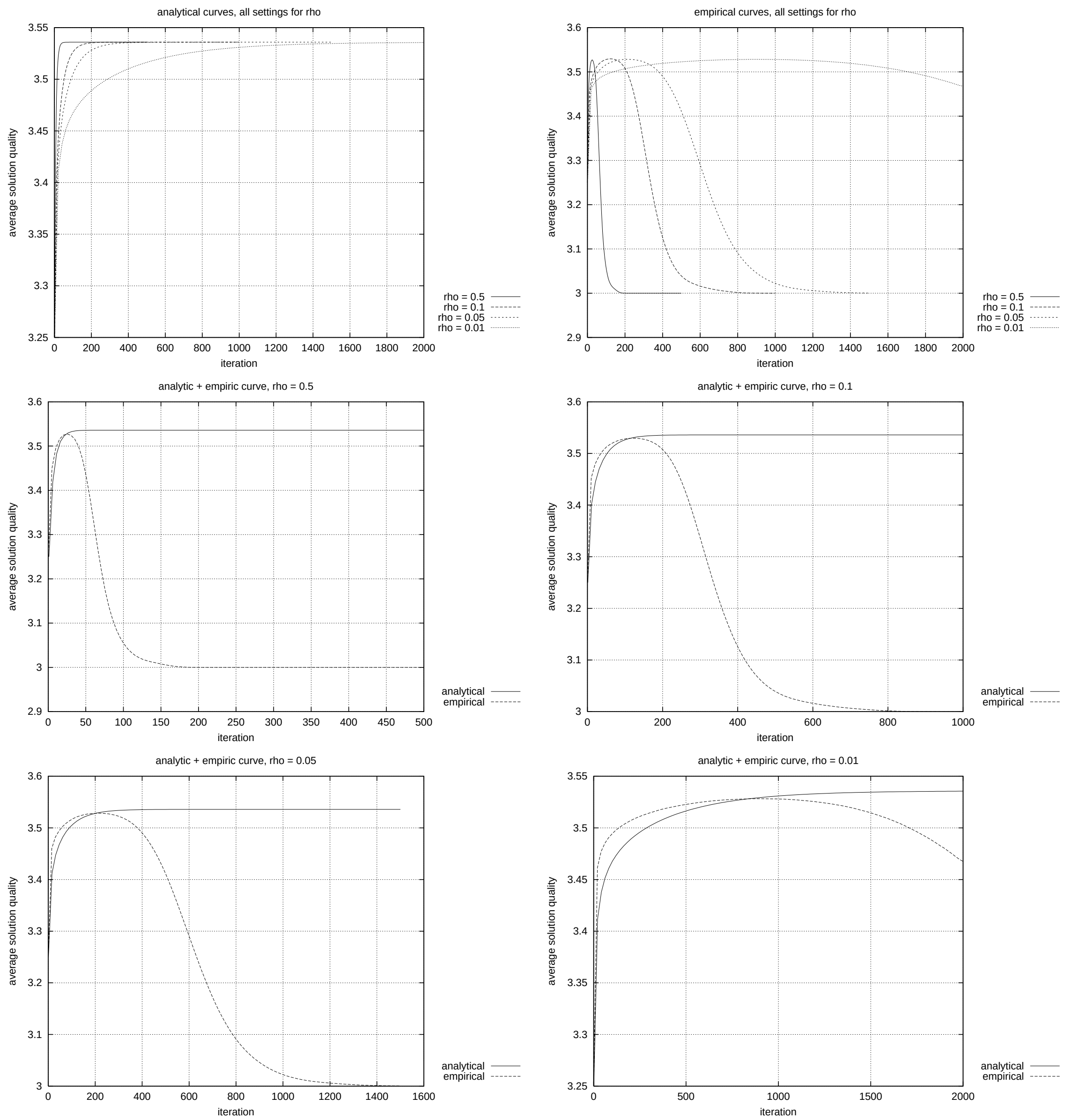

Figure 3: Performance curves for problem instance 1. The graph in the upper left corner shows the analytic curves for four different values for $\rho \in \{0.5, 0.1, 0.05, 0.01\}$. In contrast to that the graph in the right upper corner shows the empirically obtained curves for 10000 solution constructions per iteration (the curves are averaged over 100 experiments). The other four graphs show the comparison of the analytic and the empirical curve for every one of the four settings for ρ.

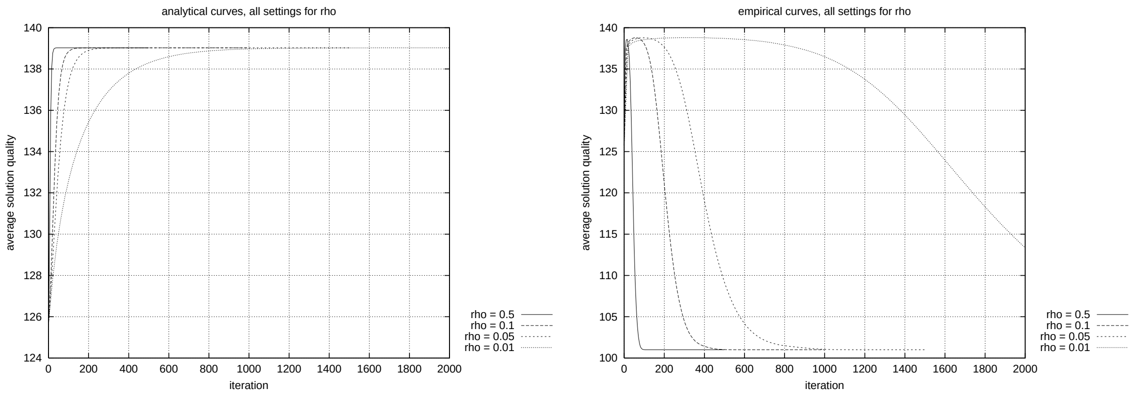

Figure 4: Performance curves for problem instance 2 (higher weights on v_2 and v_3). The graph on the left shows the analytic curves for four different values for $\rho \in \{0.5, 0.1, 0.05, 0.01\}$. In contrast, the graph on the right shows the empirically obtained curves for 10000 solution constructions per iteration (the curves are averaged over 100 experiments).

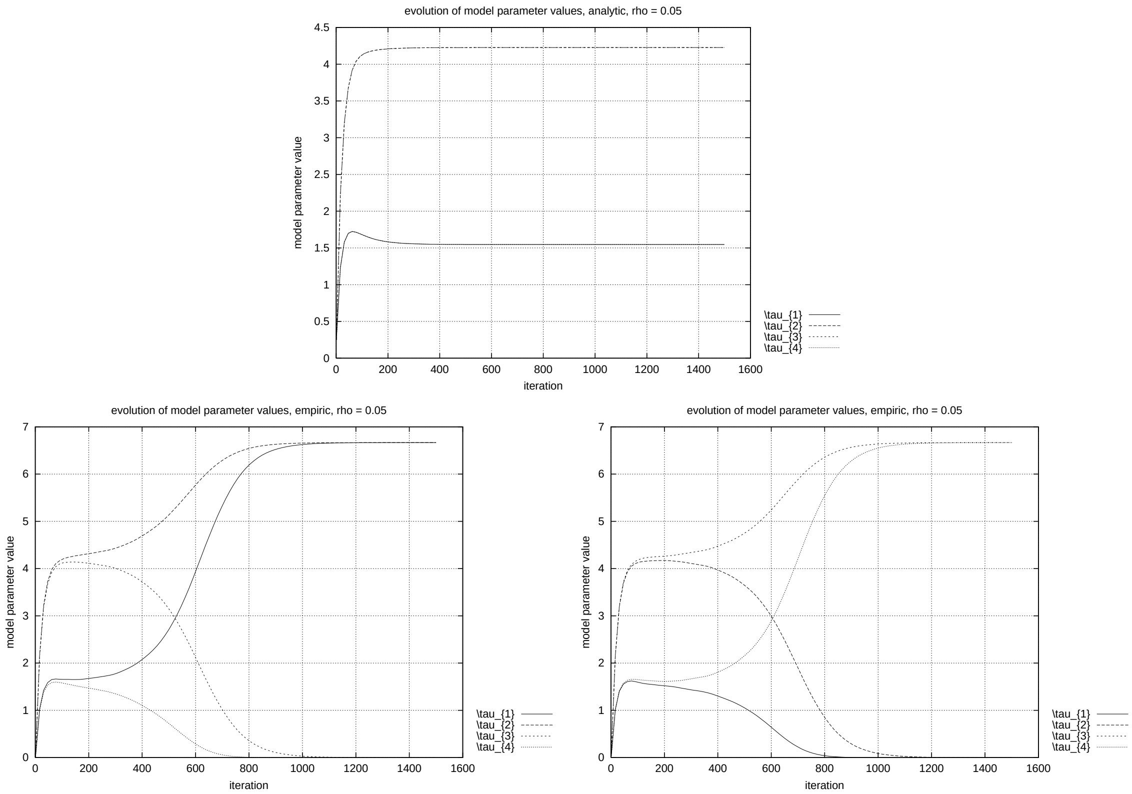

Figure 5: The three graphs show the evolution of the four model parameter values exemplary for problem instance 1, $\rho = 0.05$. The top graph shows the analytic curves (there are just two curves visible, because the curves for τ_{v_1} and τ_{v_4} (τ_{v_2} and τ_{v_3} respectively) are exactly the same). In contrast, the two lower graphs show the evolution of model parameter values empirically obtained. In the graph on the left the system is converging to solution T_a, and in the graph on the right to the solution T_c.