Solving Multiobjective Shortest Path Problems by Using Ant Colony Optimization Combined with a new Heuristic-Value-Concept - The Look-Ahead-Heuristic

Marco Fischer
mfi@hrz.tu-chemnitz.de

Sascha Häckel
shae@hrz.tu-chemnitz.de

Jens Kaminski
kaje@hrz.tu-chemnitz.de

Faculty of Economics and Business Administration
Chemnitz University of Technology
Chemnitz, Germany

ABSTRACT

The paper describes an improved method for the decision support during the selection of partners in a Virtual Enterprise. This needs a method for choosing the most capable offers (of the partners) for a customer inquiry from a pool of potential partners. The offers consist of more than one criterion and have to fulfil the tasks of a value chain. Especially on the popular field of supply chains, applicants are more and more confronted with the search for non-sequential paths which could contain parallelism. The problem arises to a Multiobjective Parallel Path Problem. Because complexity of those problems exceeds the capacities of exact procedures, the Ant Family Heuristic was developed [6]. Improving the performance in the multiobjective case the Look-Ahead-Heuristic will be introduced in this contribution. With the usage of this new heuristic, clearly better results can be achieved.

Categories and Subject Descriptors

Track [**Ant Colony Optimization and Swarm Intelligence**]: New hybrids between ACO/SI algorithms and other methods for optimization

Keywords

Multiobjective Optimization, Ant Colony Optimization, Parallel Path Problem, Hybridization

1. INTRODUCTION

Specialization and global acting are very important for the competitiveness of an enterprise in the 21st century. Thereby, the concentration on core competences implies the increase in enterprise-spanning cooperations having the objective of releasing cost reduction potentials and being present on global market places [11]. The pressure to face those challenges increases considerably especially for small- and medium-sized enterprises in order to secure their own survival. This also represents the reason for the high occurrence of supply chains or Virtual Enterprises in practice. At Chemnitz University of Technology, a virtual enterprise model has been developed in order to improve the competitiveness of small- and medium-sized enterprises.

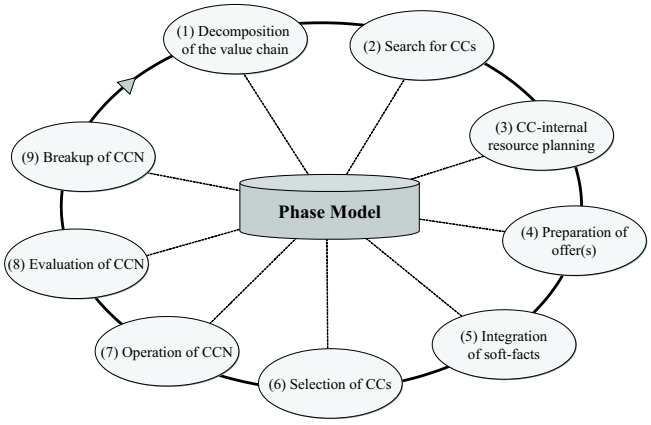

Figure 1: The Phase Model for the Genesis and Operation of a Competence Cell Network

Each net building process is triggered by a customer-inquiry for a desired product. Starting from a Process Variant Plan as a result of decomposing the value chain (see figure 1) of the product's manufacturing process, partners that are able to execute the contained process steps are searched. Therefore, a database holding a description of all participating partners is used. For getting the suitable candidates the technological feasibility is the only selection criterion at this stage. After all possible candidates are identified, they will be asked to submit an offer. The offer refers to a certain semi-manufactured product or end-product and must ensure the delivery of the whole material amount at a specified date. The partners will then schedule the required production to determine whether it is possible to deliver the inquired amount in time. If so, price, probability of delivery [12, 13] and other important criteria are submitted. Conse-

quently, a multiobjective optimization problem arises. After all inquired partners responded, the offer graph will be created. Therefore, each process step will be replaced by the corresponding partner offers. The task is to choose these orders that cover the economic objectives best.

2. PROBLEM DESCRIPTION

The resulting Parallel Path Problem is formally defined as follows [6]:

$$
\begin{aligned}
G &= (V, E) && (1)\\
V &= \{v_{ip} : i = (1, \ldots, n); p \leq n\} && (2)\\
E &= \{e_{ij} = (v_i, v_j, c_{ij}^k, g_{ij}) : v_i \neq v_j, c_{ij}^k \in \mathbb{R}^+, && (3)\\
&\quad g_{ij} \in \{1, \ldots, g_i\}, k = \{1, \ldots, K\}\} &&\\
\nexists Q &= \{(v_i, v_{i+1}), (v_{i+1}, v_{i+2}), \ldots, (v_m, v_i)\} \subseteq E && (4)
\end{aligned}
$$

The graph G contains a set of nodes (V) and a set of edges (E). Each node belongs to a nodegroup p (2) like each edge belongs to an edgegroup g (3). To rate the quality of a path, every edge contains a heuristic information c_{ij}^k for each criterion k. This could, for instance, be the path length or the production costs. All nodes of a found solution have to be connected so that every node can be reached from the source node by using the selected edges. Equation (4) expresses the condition of cycle-free tours.

The offer graph is considerably different. The occurring heuristic values and the contained path lengths of the Process Variant Plan are varying in a significant manner. This depends for example on the used material or assemblies and the utilization of the partners.

Applying optimization algorithms on Parallel Path Problems like the offer graph raises the question of how to construct a feasible solution. In a TSP-instance, a solution is a singular path that is a permutation of nodes and a subset of edges. To build a solution in Parallel Path Problems, it is necessary to respect one of each available alternative nodes in each nodegroup.

First, it is necessary to describe a few specific terms. An edge coming from a node is assigned to an edgegroup. A feasible solution has to cover the condition of containing one edge of each edgegroup of a chosen node. Like the edges, each node is assigned to a nodegroup. Nodes in the same nodegroup are representatives of complete alternatives. Because of that, a solution has to contain exactly one node out of each nodegroup. Another conclusion is that all edges of an edgegroup lead to a common nodegroup. For the practical use, quality of solutions and the aspect of how fast they are computed are very important [10]. The acceptance of a partner's offer has a far-reaching impact on a firm's production planning since it has to re-serve material and production capacity. Since the multiobjective problem is NP-complete, it should be solved approximately with an Ant Colony Optimization algorithm [7, 5]. It should enable the ants to use the new concept of the Look-Ahead-Heuristic to improve the quality of the optimization run.

3. SOLVING THE PROBLEM WITH ANT COLONY OPTIMIZATION AND LOOK-AHEAD-HEURISTIC

In the first stage of the optimization run via Ant Colony Optimization (ACO), the most important issue is to lead the artificial ants into promising regions of the solution space that will be searched through more intensely in later stages (More details in [3, 2]). However, in the beginning the pheromone values are not significant enough to have meaningful local decisions based on them. To avoid a purely stochastic search in the beginning, heuristic values are used to control the exploration. The exact specification of the used algorithm will be introduced in the last section. In the following, an approach of a new heuristic - the so-called Look-Ahead-Heuristic - for the Parallel Path Problem will be introduced. It is designed to be applied on problems that on the one hand are NP-complete in multicriteria environment and on the other hand are efficiently solvable in polynomial time for the special case of only one regarded criterion. First the well known standard method with local edgeweights will be discussed.

3.1 Initial Situation

In the standard case, local edgeweights are used as heuristic information for the ants' decision support in Parallel Path Problem as well as in Shortest Path Problems. For minimization problems, the inverted edgeweights c_{ijk}^k of the criterion k of the edge between the nodes i and j as heuristic values are used. The heuristic value of the edge, which has an influence on the probability distribution for local decisions of an ant, is thus calculated according to the following equation [8]

$$\eta_{ij}^k = \frac{1}{c_{ij}^k} \qquad (5)$$

However, if a Greedy-Heuristic based on these heuristic values constructs relatively bad solutions for the specific problem instance, a problem might also occur when using these values by ACO. The Greedy-Heuristic's solution construction is done with decisions based on solely local values. If edges with very high weights are frequently followed by edges with very low weights, such subpaths cannot be found by the Greedy-Heuristic. Especially if the solutions' path lengths vary strongly and the edgeweights scatter over a much wider range, this problem will be additionally intensified. By using these values by ACO a similar situation arises.

In the application of the Parallel Path Problem, in which an offer network should be optimized, such cases are very probable. This is due to the fact that offers not requiring any additional preliminary products cause clearly higher costs than offers offering merely an assembly of preliminary products and demand accepting several additional offers. According to this, the application of the described standard heuristic values c_{ij}^k in ACO would on average clearly prefer longer paths to shorter paths. This can cause a falsification of selection probabilities.

Therefore, it is possible to construct problem instances in which the standard method fails completely. This can even

effect on the one hand that the optimal path would be rated as the worst path and on the other hand that the worst path as the best path.

3.2 Look-Ahead-Heuristic based on Dynamic Programming

The Parallel Path Problem similar to the Shortest Path Problem is efficiently solvable with an algorithm from the field of the Dynamic Programming. Therefore, it is possible to use a Dynamic Programming-algorithm to develop Look-Ahead-Heuristik values in a pre-processing phase by which the problem of the standard heuristic value can be avoided. For this, starting from the initial node, the problem is recursively split up into subproblems. These are subsequently solved separately by backward calculation. The (partial) solutions find their way into the solution of superordinate subproblems. Analogously to Bellman's Principle of Optimality, all subpaths that can not be part of the optimal solution for criterion k, which is to optimize, can be gradually excluded [1].

The graph of a problem instance is always topologically sortable. According to this, a graph can be split up into n different stages. Every node v gets assigned the stage number which is equal to the number of edges of the longest way from the start node to the node v. Each stage m with $1 \leq m < n$ includes a finite set of subproblems which can be solved with simple decisions at this stage if all subproblems of the stage $m+1$ have already been solved. Since the decisions at stage m is depending on the objective function values of the optimal solutions of depending subproblems at stage $m+1$ it is essential that the subproblems at stage $m+1$ have been solved before. The solution of the problem has to start at the highest stage n. The set of all decisions at a stage m can be described with the system state Z_m. The optimization function transfers the system state Z_m to the next state Z_{m-1} by making all required decisions at stage m and returns the objective functions values of the subproblems to the next stage $m-1$. Since there is no dependence of the decisions at the stage m to any decisions at previous stages $i < m$ the *Markov property* is present. Therefore, all conditions that Bellman has described for the structure of dynamic programming processes are complied. Bellman's *Principle of Optimality* is as follows:

> "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision." [1]

The decomposition of a problem in n stages with a final set of subproblems and the solution of the several stages from $n, ..., 1$ is realizable with a recursion beginning at the chosen start node. The function OPTIMIZE in algorithm 1 solves instances of the Parallel Path Problem for the objective k exactly and constructs an optimal path. The split up of a problem in n steps and the solution of the single steps from $n, ..., 1$ is possible beginning with the chosen start node.

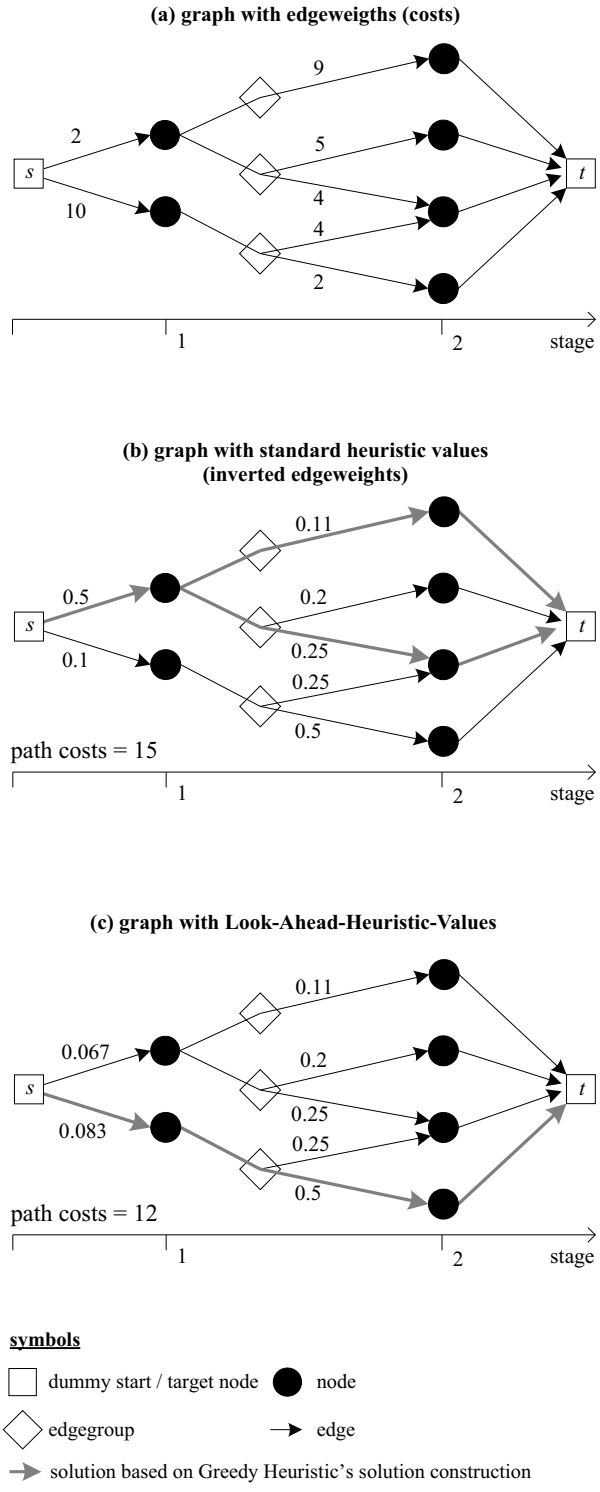

Figure 2: Comparison of standard heuristic values and Look-Ahead-Heuristic values.

The bold gray arrows show the resulting parallel path by using the Greedy Heuristic's solution construction. Such a construction is similar to the construction of the q_0 - rule in ACS in the first iterations.

Algorithm 1 Determination of the singleobjective Parallel Path Problem's exact solution with a Dynamic Program by the function OPTIMIZE and calculation of Look-Ahead-Heuristic values within the scope of a pre-processing phase by using the function BUILDHEURISTIC.

```
1:  function BuildHeuristic(start node s, target node t)
2:    foreach objective k
3:      Optimize(solution, s, t, k)
4:      foreach edge e_ij in graph
5:        heuristicvalues[e_ij]← heuristicvalues[e_ij] + solution[j]
6:      end for
7:    end for
8:  end function

9:  function Optimize(path[] solution, node n, node target, objective k)
10:   if n marked
11:     solution [n] ←empty
12:     if n ≠ target then
13:       foreach edgegroup group ∈ n
14:         foreach edge e_nj ∈ group
15:           subpaths [e_nj] ←Optimize(j, k)
16:           Add edge e_nj to subpaths [e_nj]
17:         end for
18:         Add all edges of that path with best objective function value f^k in subpaths to solution [n]
19:       end for
20:     end if
21:   end if
22:   Mark node n
23:   return solution [n]
24:  end function
```

$$\eta_{ij}^k = \frac{1}{c_{ij}^k} + f^k(j)^* \tag{6}$$

With the help of this method an optimal subpath is calculated for each node v to a target node with the appropriate objective function values $f^k(v)*$. From these information, a Look-Ahead-Heuristic-value for all edges in the graph can be calculated according to the equation (6) for minimizing problems. This is shown by the function BUILDHEURISTIC in algorithm 1.

4. SPECIFICATION OF THE APPLIED ALGORITHM

For the evaluation of the new heuristic values a simple Ant Colony algorithm was used deliberately to illustrate how the solution quality can vary strongly depending on the used heuristic. For this, the basic version Ant Colony System as a multiobjective algorithm in combination with the Ant Family Heuristic was used. The evaluation was based on biobjective problems. The heuristic's aim is to determine the Pareto-front of the problem instance.

4.1 Ant Colony System

The algorithm is based on the Ant Colony System [4]. One heterogeneous colony whose ants look for solutions regarding their importance of the different criteria was employed. For a biobjective optimization problem the i-th ant of the colony with the size s has the following weights, whereas $i \in \{1...s\}$ is. [9]

$$w_i^0 = \frac{i-1}{s-1} \text{ and } w_i^1 = 1 - w_i^0 \tag{7}$$

Simply the artificial ants that have constructed a solution that lies in the non-dominated front may perform a pheromone update. To ensure the same importance of each iteration, the approach described by Merkle was used and extended by an additional weight w_k [9]. Each ant is allowed to modify the pheromone values of the criterion k for all edges of its path about the value

$$\Delta\tau_{ij}^k = \frac{1}{m} \cdot w^k \tag{8}$$

whereas m is the number of one iteration's ants that can perform an update, and w_k is the importance of the criterion k to the ants. K is the total number of objectives the problem has. For local decisions the Pseudorandom Proportional Rule is used [4]. The probability distribution for the selection of local decisions is made according to equation (9), whereas g illustrates the edgegroup to which all edges e_{ij} belong that are alternatively available in the local decision.

$$p_{ij} = \frac{\sum_{k=1}^{K} w^k \cdot \left(\eta_{ij}^k\right)^\alpha \cdot \left(\tau_{ij}^k\right)^\beta}{\sum_{e_{ij} \in g} \sum_{k=1}^{K} w^k \cdot \left(\eta_{ij}^k\right)^\alpha \cdot \left(\tau_{ij}^k\right)^\beta} \tag{9}$$

The values α and β serve the control of the influence of pheromone and heuristic values. To antagonize a too fast convergence of the method, relatively high initial values are chosen for the pheromone. The evaporation is performed after each iteration with a steady factor $\rho \in [0, 1]$.

4.2 Ant Family Heuristic

Since in the Parallel Path Problem not only singular paths have to be determined, the approach of the Ant Family Heuristic is additionally used [6]. This heuristic describes the construction of feasible solutions in Parallel Path Problems by introducing an ant-replication strategy for ACO. In the Ant Family Heuristic, each ant has the ability to create clones of itself for each path that has to be visited. This is necessary if the leaving edges of the current node belong to more than one edgegroup. So for each node-group one clone will be created. All clones and the original ant exchange information by sharing a common family mind. After the cloning, each individual chooses one node of its delegated edgegroup by applying the decision rule of the underlying base-algorithm. In case of the convergence of two or more paths meeting at a common nodegroup additional conditions have to be considered. An ant that has to select an edge of an edgegroup has to look up in the family mind whether the edge-group's target nodegroup has already been visited by another family member. If so, the ant takes the edge between its current position and the selected node of the next nodegroup. After that, the clone is no longer useful for the construction process and will be eliminated. If the nodegroup has not been visited, the ant's task is to continue the construction process. The solution construction ends if the only left family member's current position is at a leafnode of the graph.

5. RESULTS

Within the scope of the evaluation, problems were examined that are also solvable multiobjectively with an algorithm of the dynamic programming at appropriate time in order to calculate the correct Pareto-optimal front. This way, it was possible to examine the ACO's behavior in detail. The underlying problem is about a biobjective problem with a tree of 17050 nodes and edges apiece, whereas the edges are arranged to 5664 edgegroups as a total. The solution space covers $1.4 \cdot 10^{24}$ solutions of which 242 are Pareto-optimal. The local edgeweights c_{ij}^0 and c_{ij}^1 for the objectives were randomly distributed in [100..1000] and [0..100], respectively. The value q_0 for the *pseudorandom proportional rule* was set with 0.05, the evaporation rate with $\rho = 0.1$. The colony size s was set to 20 ants and the parameters α and β were both set to 1.

In Figure , the several results are presented. The same base-algorithm with the same parameter settings was used in the application of the standard heuristic values as well as in the application of Look-Ahead-Heuristic-values. The homogeneous run of the $\lambda - branchingfactor$ [4] illustrates that the algorithm's behavior with different heuristic values is fundamentally similar Figure part b. The initial value is based on the fact that between 3 and 5 alternatives are available to each ant at one local decision. The basic difference between the methods is shown both by the aver-age solution quality of the found non-dominated solutions and in the size of the non-dominated front. The average distance of the

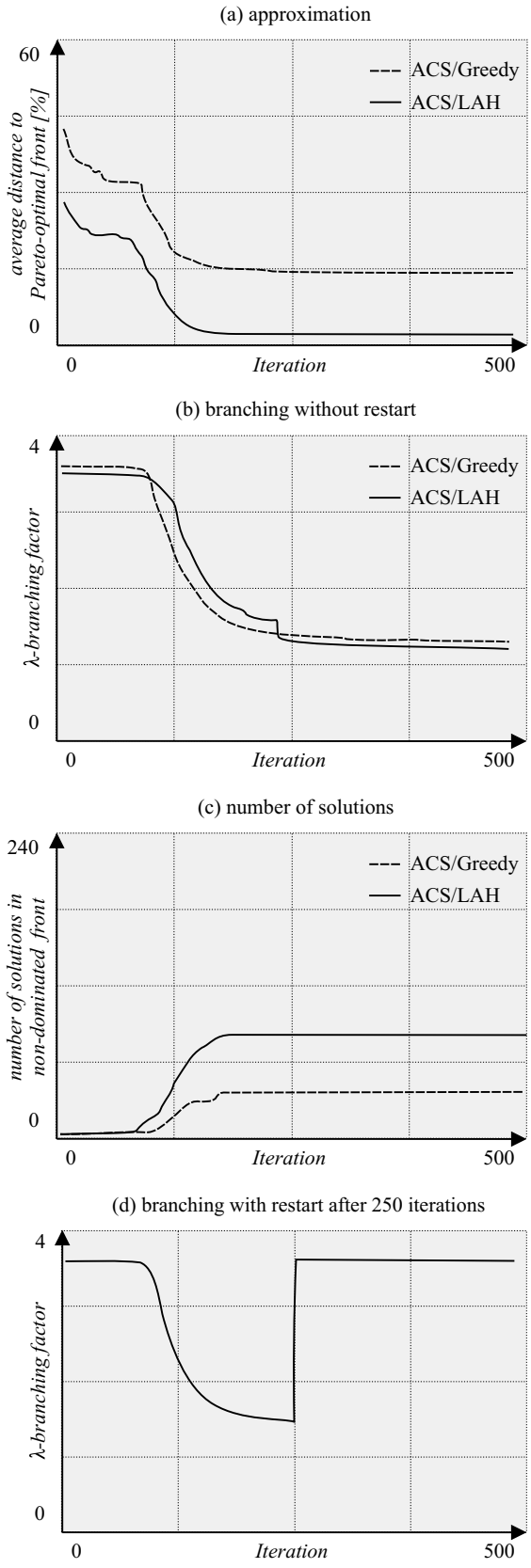

Figure 3: Some Characteristics of the Ant Colony Optimization with the usage of the standard heuristic values and the Look-Ahead-Heuristic.

found non-dominated front declines from ca. 15% to ca. 2% after the convergence of the algorithm. The size of the non-dominated front almost doubles from 43 to 85 solutions. Since in the underlying problem 242 solutions belong to the Pareto-front, the front sizes are relatively small. However, this can be ascribed to the fast convergence of the ACS-method on the one hand and to the applied update strategy on the other hand, because in Figure 1 it becomes apparent that also restart strategies do not entail any further improvements. Therefore, the behavior of an ACO algorithm can be clearly improved by the application of Look-Ahead-Heuristic-values since both larger non-dominated fronts and a much better approximation of the Pareto-front can be achieved. Additionally, all marginal solutions of the Pareto-front, which are exclusively optimal regarding one objective, can be determined by the application of the dynamic program.

6. CONCLUSION AND OUTLOOK

The Look-Ahead-Heuristic present a very promising alternative for the solution of the Multiobjective Parallel Path Problem as well as the Shortest Path Problem. Very good results were already achieved by using a very simple, fast converging ACS algorithm. This allows to conclude that the application of the Look-Ahead-Heuristic in combination with ACS has a clearly higher efficiency than the application of the standard heuristic information combined with ACS. In order to improve the behavior of the optimization further and to avoid a too fast convergence, the Look-Ahead-Heuristic should be applied with a MAX-MIN-Ant System [4] and under the use of several colonies. Using restarts, it was furthermore observed that the update strategy is in need of improvement as well. Since singleobjective problems are efficiently solvable with the Dynamic-Programming-algorithm, it is also possible to determine upper and lower limits for the single objective function values of a criterion. This way, scales can be derived on which solutions can be rated by percentage regarding their quality. As a result of this rating, an update strategy is possible with which the exploration and exploitation of the optimization run by artificial ants can possibly be controlled in a much higher degree than at the application of the present strategy.

7. ACKNOWLEDGMENTS

This work has been supported by the Collaborative Research Center 457.

8. REFERENCES

[1] R. E. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, New Jersey, 1957.

[2] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence - From Natural to Artificial Systems.* Oxford University Press, 1999.

[3] M. Dorigo and G. DiCaro. *The Ant Colony Optimization Meta-Heuristic. In New Ideas in Optimization*, pages 11–32. McGraw-Hill, 1999.

[4] M. Dorigo and T. Stützle. *Ant Colony Optimization.* Massachusetts Institute of Technology, 2004.

[5] M. Ehrgott. *Multicriteria Optimization.* Springer-Verlag, Heidelberg, 2. edition, 2005.

[6] M. Fischer, T. Giese, and H. Jähn. Optimization in production networks in case of tree structures within value chain by using a new ant colony approach. In *Proceeding of the 20th International Conference on CAD/CAM, Robotics and Factories of the Future*, San Cristobal, Venezuela, 21.-23. Juli 2004.

[7] M. Fischer, T. Giese, and H. Jähn. Optimizing the selection of partners in production networks. *International Journal of Robotics and Computer-Integrated Manufacturing*, 20(6):593–601, 2004.

[8] M. Guntsch. *Ant Algorithms in Stochastic and Multi-Criteria Environments.* Karlsruhe, 2004.

[9] D. Merkle. *Ameisenalgorithmen - Optimierung und Modellierung.* Karlsruhe, 2002.

[10] W. Mülder. Electronic data interchange - standard mit neuen perspektiven. *Office Management*, 1, 1999.

[11] C. K. Prahalad and G. Hamel. The core competence of the corporation. *Harvard Business Review*, 68:79–93, 1990.

[12] T. Teich. *Extended Value Chain Management - ein Konzept zur Koordination von Wertschöpfungsnetzen.* Verlag der GUC, 2003.

[13] T. Teich, L. Zschorn, and H. Jähn. Management of production networks - a new approach to work with probabilities of delivery. In *Proceedings of the 12th International Conference on Flexible Automation & Intelligent Manufacturing - FAIM 2002*, pages 762–771, 2002.