Multiagent Matching Algorithms with and without Coach

Frieder Stolzenburg* — **Jan Murray**** — **Karsten Sturm*****

* *Hochschule Harz, Univ. of Applied Studies and Research, Automation and Computer Sciences Dep., Friedrichstr. 57–59, D–38855 Wernigerode, fstolzenburg@hs-harz.de*

** *Univ. Koblenz-Landau, Campus Koblenz, Computer Science Dep., Artificial Intelligence Research Group, Universitätsstr. 1, D–56070 Koblenz, murray@uni-koblenz.de*

*** *Landesmedienzentrum Rheinland-Pfalz, Hofstraße 257c, D–56077 Koblenz, sturm@lmz.bildung-rp.de*

ABSTRACT. *The concept of "agent" has been used to describe many different artefacts: software programs, mobile robots, or even human beings. In a system consisting of many agents, negotiating and interacting with each other, the control of the behavior of the overall system becomes difficult. How the overall system behavior changes with respect to distributed rational decision making as a function of different utility measures is also problematic. This paper uses a well-known matching problem to discuss options for control in a multiagent scenario. In particular, we discuss several different matching criteria and several different algorithms for resolving the matching problem.*

RÉSUMÉ. *Le concept d'« Agent » est utilisé pour décrire des objets très différents : programmes informatiques, robots et même des agents humains. Dans un système qui comprends plusieurs agents, en négociation et interaction les uns avec les autres, le contrôle du comportement du système devient difficile. La façon dont le comportement du système est affecté par certaines situations de prise de décision distribuées en fonction de mesures d'utilité différentes est aussi problématique. L'article utilise un problème de allocation des formes très classique pour envisager les possibilités de contrôle dans un scénario multi-agent. En particulier, nous discutons un certain nombre de critères de allocation et présentons des algorithmes pour résoudre le problème de allocation.*

KEYWORDS: *Decision Making; Matching; Multiagent Systems; RoboCup.*

MOTS-CLÉS : *prise de décision, allocation, système multi-agent, RoboCup.*

JDS – 15/2006. Decision Support Systems, pages 215 to 240

1. Introduction

A problem that is frequently encountered in computer science and other contexts is to find a mapping of the elements of one set to the elements of another one satisfying some given constraints. These *matching problems* have been studied extensively, and a number of matching algorithms for different needs have been proposed throughout the years, *e.g.* for the well-known *stable marriage problem* (Gale *et al.*, 1962), which find matchings between two sets based on preference lists or rankings. Many different real-world matching tasks such as man-to-man marking in (simulated) robotic soccer, public transportation problems, etc. can be seen as instances of the abstract matching problem.

In a multiagent scenario, *i.e.* in a setting where at least one of the matching sets represents a group of autonomous agents, a number of interesting facets is added to the general matching problem. First of all, the decision and matching process may then be centralized or decentralized. In the former case, one distinguished agent, called *coach*, calculates a matching for the whole group and then informs the other agents of the generated matching. Alternatively, a matching can be computed in a decentralized (or local) way, if all agents participate actively in the process. In this case, an agent communicates with some or all of its mates to calculate an appropriate matching. Both, the centralized and the decentralized method have advantages and drawbacks. A central control instance may always be a bottleneck and a potential point of failure in a system. A decentralized approach, however, may be infeasible because of time constraints or high communication costs. Another interesting scenario wrt. decision systems arises, if *both* matching sets represent groups of agents, that do not necessarily have the same interests or preferences for a matching. Such a scenario may lead to a variety of the stable marriage problem (Gale *et al.*, 1962), as a solution that satisfies the rankings of the members of both groups as good as possible has to be found.

However, in this paper we concentrate on settings where one matching set consists of agents and the other represents passive entities (which may also be agents, that are not actively participating in the decision making process). Here, we examine matchings that are based upon distances in the d-dimensional Euclidean space $\mathbb{R}^d$, especially $\mathbb{R}^2$, *i.e.* in the plane, as distance-based matching problems frequently arise in real-world applications. In Section 2, three possible application scenarios are presented, namely man-to-man marking in robotic soccer (RoboCup), public transportation problems, and rescue actions in a disaster area. After that, we introduce basic definitions and discuss related work on matching and coaching in multiagent systems in Section 3.

Section 4 refines various criteria for computing (distance-based) matchings formally and examines the relations among them. After that, algorithms for decentralized matching and globally maximal matchings (where the maximal distance between agents and opponents is minimized not only for the whole set of elements but also for each submatching) are given. The decentralized algorithm (Section 5) determines a possibly partial matching between two sets, which locally satisfies one of the match-

ing criteria. This algorithm only requires communication among pairs of agents. The algorithm for computing a globally maximal matching between two sets by means of the coach agent (Section 6) extends the one presented by Efrat *et al.* (2001). One matching algorithm has been implemented in the *RoboLog Koblenz* simulated soccer team. It is an algorithm for computing a (ranking-based) stable marriage (Murray *et al.*, 2002; Sturm, 2003), also called globally minimal matching here, making use of the coach agent that computes matchings in order to optimize opponent marking. In stable matchings, the minimal distance between agents and opponents and hence the corresponding approaching time is minimized. In Section 7, we briefly address the need for non-geometric matchings in the soccer scenario. Section 8 finishes the paper with some concluding remarks.

2. Application scenarios

As already mentioned, the matching problem occurs in many different application scenarios. In the following, we briefly discuss some of them. The variety of applications shows that there is a need for both centralized and decentralized matching procedures for decision making in multiagent systems, *i.e.* with and without coach.

2.1. *RoboCup and simulated soccer*

The RoboCup Initiative aims at fostering research in robotics, artificial intelligence, and multiagent systems. As an example domain *robotic soccer* has been chosen, because soccer combines many interesting problems, *e.g.* dealing with uncertain and incomplete information, cooperation and coordination in a team of autonomous agents, decision support in multiagent systems, or planning and acting in a highly dynamic environment. RoboCup is divided into several leagues, which focus on different research aspects. The *simulation league* deals with aspects of situated multiagent systems like teamwork, spatial reasoning, decision making, and opponent modeling.

In the RoboCup Simulation League, two teams of 11 autonomous agents compete in a simulated soccer match. The two-dimensional (2D) discrete-time simulation is carried out in a client/server style by the *Soccer Server* (Chen *et al.*, 2003). The Soccer Server maintains a model of the world and all objects on the field. Sensory input simulating a noisy vision system is sent to the agents at regular intervals. In each simulation step, clients may send commands to the Soccer Server, which are taken into account by the simulator when the world model is updated for the next step.

The agents forming the actual team are almost identical to one another. With the help of different *player types* the agents' capabilities, *e.g.* the maximum strength of a kick, are parameterized. The *coach agent*, however, is different. It gets global and noiseless information from the Soccer Server but it cannot physically interact with its environment. It can only support its team by giving advice or information to its players. To prevent the coach from controlling its team in a (too) centralized fashion,

its messages reach the players with a substantial delay if the ball is in play. In addition, the number of messages a coach can send during the game is limited.

Obviously, man-to-man marking in soccer and geometric matching are closely related. Therefore, in the RoboLog soccer simulation team, the coach employs a global ranking-based algorithm for computing a stable matching (called globally minimal matching here). With this algorithm, players are assigned to opponents they have to mark (Murray *et al.*, 2002; Sturm, 2003). Man-to-man marking requires to compute a mapping, such that all agents reach the opponent positions as quickly as possible. Hence, matching where the maximal distance between agents and opponents is minimized not only for the whole set but also for each submatching (called globally maximal matching here) seems to be the most appropriate procedure (see Section 4.2).

2.2. *Public transportation*

As another application scenario, we consider a taxi service in a big city. At a given time the taxis are spread throughout the city. From various spots in the city, customers call the headquarters to order a taxi. The headquarters, working as a match-maker in this decision making process, then has to assign each free taxi to a waiting customer in such a way that the customers are satisfied with the service. Clearly, this is a setting in which a centralized matching procedure should be preferred because of the time bounds. A decentralized matching would require the taxi drivers to first spread the available information among themselves and then find a matching, which usually takes a substantially longer time than the centralized approach. In addition, because of the online nature of this problem, suitable matchings should be recomputed as soon as possible, whenever a new customer shows up.

2.3. *The rescue scenario*

In the unfortunate event of a major disaster (*e.g.* an earthquake), extensive rescue actions have to be taken as soon as possible. The disaster area has to be searched, injured or buried people must be found and saved, fires have to be extinguished. One severe problem is that the disaster area still holds lots of dangers for human rescue squads, so having decision support from autonomous robots is strongly desired.

The tasks of robots in this scenario include exploring buildings, finding buried or injured persons to establish contact to them, and providing information to the human rescue forces. While this is a task that is usually controlled from a central headquarters, it may very well be that the communication lines break down, so that the headquarters are out of reach. But (a subset of) the rescue robots may still be able to reach one another by radio. As time is of utmost importance in such a rescue scenario, it would be desirable that the robots are able to coordinate and go on with their exploration tasks until the connection to the headquarters is re-established. Therefore, they

must be able to find a mapping between robots and target locations in a decentralized way. There is also a so-called rescue league in the RoboCup (see also Section 3.2).

3. The geometric matching problem and prior work

Let us now introduce some formal notation and formalism for the matching problem. The most interesting case in our context is *maximal matching*, where the maximal distance between agents and opponents is minimized. This appears to be best-suited for the application scenarios just mentioned, because then all distances between agents and opponents, also the maximal ones, are not overly long. But beforehand, we have to introduce the concept of matchings in general (Definition 3.1) and optimality criteria for them (Definition 3.3 and 3.4).

DEFINITION 3.1 (Matching) Let $P = \{p_1, \ldots, p_{n_1}\}$ and $Q = \{q_1, \ldots, q_{n_2}\}$ be two sets of agent positions in $\mathbb{R}^d$, where d is the number of dimensions in the Euclidean space. A *partial matching* M is a set of (undirected) edges between P and Q, such that each vertex from $P \cup Q$ has at most one edge incident in M. If M is of maximal cardinality, it is called a *complete matching* or just *matching* for short. A subset $M' \subseteq M$ is called a *submatching* of M. We also make use of the following notation: if M is a matching with $\{p, q\} \in M$, then $d(p, q)$ denotes the distance between p and q, and we write $d_M(p)$ for $d(p, q)$.

Usually, P and Q have the same cardinality n (*i.e.* $n_1 = n_2 = n$), and the most interesting case is where the agent positions are points in the plane (*i.e.* $d = 2$), although there are already autonomous robots working in three dimensions (*i.e.* $d = 3$), *e.g.* an industrial robot arm or a (simulated) flying robot. In our context, P represents positions of agents that all have to reach one of the positions in Q as quickly as possible. As distance $d(p, q)$ between two points $p \in P$ and $q \in Q$, we simply take their Euclidean distance (induced by the L_2 norm).

DEFINITION 3.2 (Euclidean distance) The *Euclidean distance* $d(p, q)$ between the points $p = (p_1, \ldots, p_d) \in P$ and $q = (q_1, \ldots, q_d) \in Q$ in $\mathbb{R}^d$ is

$$\sqrt[2]{(p_1 - q_1)^2 + \cdots + (p_d - q_d)^2}$$

which is the corresponding L_2 norm of the vector $p - q$.

3.1. *Criteria for matchings*

In the following, we state some definitions of evaluation criteria for matchings. Criteria such as stability, Pareto efficiency and social welfare are standard in the literature on multiagent systems and distributed rational decision making (Moulin, 1988; Sandholm, 1999). We adapt them here to the (geometric) matching problem. A well-known optimality criterion for matchings is stability (Gale *et al.*, 1962; Irving *et al.*, 1987):

DEFINITION 3.3 (Stable matching) A matching M between P and Q is *stable* iff for all pairs $\{p_i, q_i\} \in M$ and $\{p_j, q_j\} \in M$ (with $i \neq j$), we have $d(p_i, q_j) \geq \min(d(p_i, q_i), d(p_j, q_j))$, *i.e.*, there is no pair $\{p_i, q_j\} \notin M$ where both agents prefer each other over their current partners (because they are closer to each other). Therefore, M is also called a *stable marriage* (in this context with distance-based ranking).

General evaluation criteria

Unfortunately, stability is usually not enough for real world applications. Some other kind of optimality criterion is needed. In Figure 1a, for example, the stable (distance-based) matching is indicated with dashed lines. But in most applications the solution indicated by the solid lines would be preferred, because the sum of lengths (5 *vs.* 7) and the maximal distance (3 *vs.* 6) is better.

We observe that the dashed solution is also the only ranking-based stable matching, if we just consider simple rankings as in Gale *et al.* (1962) and Irving *et al.* (1987), derived from the concrete distances. For this, all possible distances between points in P and Q must be pairwise different. Incidentally, the set of distance-based stable matchings always coincides with the set of such ranking-based stable matchings. Let us now introduce other interesting properties of matchings.

DEFINITION 3.4 Let M and M' be matchings between P and Q. Then, M is called

1) *minimal* iff $\min_p d_M(p) \leq \min_p d_{M'}(p)$, *i.e.*, the minimal distance is minimized,

2) *maximal* iff $\max_p d_M(p) \leq \max_p d_{M'}(p)$, *i.e.*, the maximal distance is minimized,

3) *Pareto efficient* iff $d_{M'}(p_1) < d_M(p_1)$ for some $p_1 \in P$ implies $d_{M'}(p_2) > d_M(p_2)$ for some $p_2 \in P$, *i.e.*, nobody can be better off unless at least another one is worse off, and

4) *optimal* iff $\sum_p d_M(p) \leq \sum_p d_{M'}(p)$, *i.e.*, the sum of distances is minimized,

for all matchings M' between P and Q. Note that, in this context, P and Q consist of the respective elements occurring in M.

Related work in economics and computer science

In economics, *welfarism* takes social welfare to be a function of (and only of) individual utilities (corresponding to small distances in our context). The utilities can be combined in different ways: *e.g.* by adding, as under *utilitarianism*, or by concentrating on the utility of the worst-off agent, as under *egalitarianism*, where it is tried to equalize the individual agent utilities (Moulin, 1988). An optimal matching (according to Definition 3.4) clearly optimizes the *social welfare*, understood as the sum of all agents' payoffs, and hence implements utilitarianism. In contrast to this, a maximal matching, also called *bottleneck matching* (Efrat *et al.*, 2001), optimizes the utility level of the agent who is worst off, which therefore implements egalitarianism.

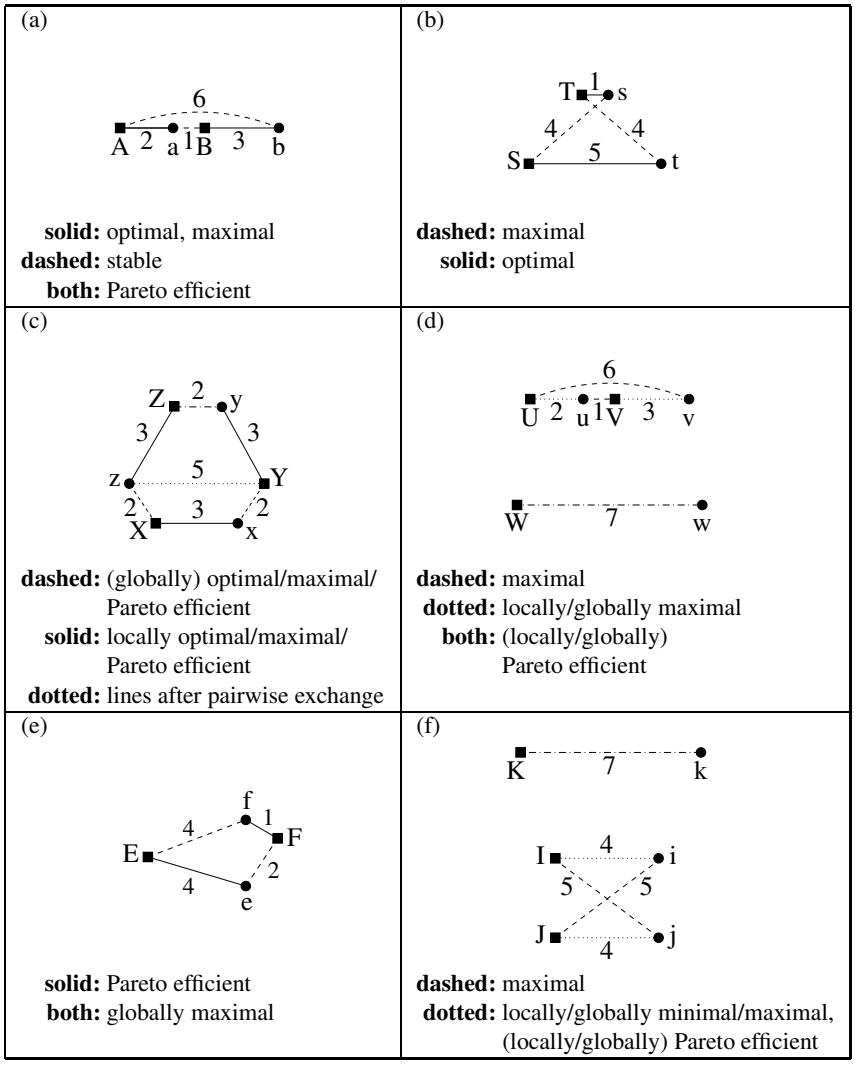


Figure 1. *Matching (counter)examples. Boxes denote elements of P, circles stand for elements of Q. Only relevant properties of the solutions are given. The respective distances are annotated at the connecting lines. Different line styles (solid, dashed, or dotted) indicate different matchings whose properties are summarized. Dash-dot lines count for both the dashed and dotted solutions*

Pareto efficiency is also a well-known property studied in economics and competitive multiagent systems and hence in distributed rational decision making (Sandholm, 1999). Moulin (1988) also considers a weaker notion of Pareto efficiency, where not all agents can properly improve their utility at the same time, defined as follows. A matching M is *weakly Pareto efficient* iff, for all matchings M' between P and Q, it holds $d_{M'}(p) \geq d_M(p)$ for some $p \in P$. Unfortunately, none of these properties are equivalent in general, as Figure 1a and 1b demonstrate, but see Sect. 4 for implication relationships among the evaluation criteria.

For some of these settings, there are many known algorithms in the computer science literature. For instance, Agarwal *et al.* (1999) provide an $O(n^{2+\epsilon})$ algorithm for the optimal (utilitarian) matching problem (called *minimum-weight bipartite Euclidean matching* there). Bottleneck matching is considered in Efrat *et al.* (2001), presenting an algorithm in $O(n^{1.5} \log n)$ time. See also Section 3.3.

This paper deals with the development of specialized matching algorithms. In addition, we address the problem of finding matchings in a multiagent context where decentralized procedures may be required. Thus, the idea of this paper is trying to combine both aspects by proposing different multiagent matching evaluation criteria and algorithms. There is a lot of related work in each of these fields, but not on their combination. Hence, we will discuss both aspects separately in the next two subsections.

3.2. *Centralized and decentralized matching in the RoboCup domain*

In this section we present some related work from practical applications in the RoboCup domain. The RoboCup simulation leagues are targeted at research in the fields of multiagent coordination, cooperation, and decision making. This is achieved in two domains, namely robotic soccer and the so-called rescue simulation. The soccer domain consists of several sub-competitions. We will present some related work from the *2D soccer competition*, the *soccer coach competition*, and the *rescue simulation*.

2D soccer competition

Within the 2D soccer simulation various approaches to agent coordination and team play have been developed. The team *UvA Trilearn* (Kok *et al.*, 2003) makes use of *coordination graphs* for coordinating a team of agents. With the help of those graphs certain set plays can be hard-wired within an agent. All agents participating in such a set play have the same knowledge about the world and follow the same set of coordination graphs, so they implicitly execute the same play.

Although the use of a specialized coach agent is permitted during games in the 2D soccer league, this is rarely done. Almost all teams use the online coach for substituting the default player type for heterogeneous players at the beginning of a match, but only few teams employ the coach during the game. The team *Virtual Werder* tried to determine the formation of the opponent team and changes thereof. Based on this

information, the formation Virtual Werder used during the match was adjusted (Visser *et al.*, 2001).

Soccer coach competition

The coach competition focuses on opponent modeling and learning from observations. In this competition the coach agent has to improve the playing ability of a team against a fixed opponent by giving advice using the standard language *Clang* (Chen *et al.*, 2003). Previous matches of the opponent may be analyzed by the coach in order to get useful information about the team for further decision support. As the coach has to realize most of the team strategies and the number of utterances it may make during a match is limited, participants usually focus on controlling more global aspects of the team, *e.g.* formations and general strategy. Kuhlmann *et al.* (2005) use machine learning to extract rules for setting up formations and offensive and defensive actions for the coached team. The *FC Portugal coach* (Reis *et al.*, 2004b) gives advice based on statistical analysis of previous games of the opponent. In Riley *et al.* (2002), formation learning and set-play selection, in which the coach uses a model of the opponent to direct the players to execute a specific plan, is considered among others.

Rescue simulation

In the RoboCup rescue simulation a disaster in a major city (maps include parts of Kobe/Japan and Foligno/Italy) is simulated. A team of heterogeneous agents consisting of ambulance teams, police forces, fire brigades, and headquarters (HQ) for each of these types has to cooperate in order to minimize the number of fatalities and destroyed buildings in the city.

The *FC Portugal Rescue Team* (Reis *et al.*, 2004a) adapts strategies developed for the successful *FC Portugal* soccer team, *e.g.* situation based positioning of agents, for use in the rescue scenario. Agents of the same type are coordinated in a centralized way by the corresponding HQ agents. Paquet *et al.* (2004) follow an interesting approach in the team *DAMAS-Rescue*. For each of the three agent types, a different control strategy is used. Ambulance teams are controlled in a strictly centralized way by their HQ agents. For fire brigade agents, a mixture of centralized and decentralized control is used. The fire brigade HQ assigns different buildings and blocks to the individual agents. Within these blocks the fire brigade agents have to coordinate in a decentralized way to extinguish the fire. Police agents, finally, are assigned to a zone in the disaster area based on a simple distance based matching at the beginning of the simulation. After that they act autonomously, *i.e.* without any control by the central police HQ agent.

3.3. *The matching problem in the literature*

Let us first review the field of matching in computer science. A matching in an (undirected) graph G is a set of edges, that are two-element subsets of the vertices, such that there are no two edges sharing a vertex. In this context, we are interested

in weighted matching, where each edge is associated with a weight $w \geq 0$. G may be assumed to be a complete bipartite graph, *i.e.*, there is an edge for every pair of vertices from two disjoint sets P and Q. In addition, we are interested mainly in complete matchings, *i.e.* of maximal cardinality (see Definition 3.1). An excellent new textbook containing a lot of material on graph theory and matching was written by Jungnickel (2005).

Stable marriage

One of the earliest formulations of matching was the problem of *stable marriage*. In this problem, we have a set of n men and n women. Each person has their own ranking list of the persons they want to marry. The weights w may be non-symmetric in this case, *i.e.*, a man and a woman may rank each other differently. The job is to determine an assignment of n stable marriages, such that there is no dissatisfied non-married pair, who prefer each other over their current partners (see Definition 3.3). Gale *et al.* (1962) show that a stable marriage is always possible for all persons, and the solution can be found in $O(n^2)$ time. Irving *et al.* (1987) improve the procedure by looking for the optimal solution among the stable (ranking-based) matchings, where the rankings are summed up as in the Borda voting protocol (Moulin, 1988). The optimal solution can be found in $O(n^4)$ time, as demonstrated by Irving *et al.* (1987). A ranking-based algorithm for computing stable matchings (also called globally minimal matching here) was integrated in the man-to-man marking procedure of the RoboLog team (see Section 7 and Sturm (2003)).

The general matching problem

In the literature, mainly optimal (utilitarian) and maximal (egalitarian) matching are considered. The optimal matching problem corresponds to the *assignment problem*, *i.e.* weighted bipartite matching where the sum of the (not necessarily geometric) edge lengths has to be minimized. Kuhn (1955) states the so-called *Hungarian method* which correctly solves the problem for a complete bipartite graph with $2n$ vertices in $O(n^3)$ arithmetic operations (see also Papadimitriou *et al.* (1998, Theorem 11.1)). Here, the problem is reduced to a linear program, subject to some constraints. The most interesting matching criterion in our context is maximality, where the maximal weight is minimized, *i.e.* egalitarianism. This leads to what is called *bottleneck matching*. It is not considered as often as optimal (utilitarian) matching in the literature, but see *e.g.* Efrat *et al.* (2001), Papadimitriou *et al.* (1998), or Jungnickel (2005). Globally maximal matching, where the maximal distance between agents and opponents is minimized not only for the whole set of elements but also for each submatching, as introduced in this paper, is even more useful in practice than simple maximal matching (see Section 4.2).

Geometric matching and related problems

If the weights are geometric, *i.e.* Euclidean distances, then the matching algorithms can be improved. Avis (1983) reviews (among others) results on heuristics for

matching problems where the vertices are points in the plane, providing worst-case ratio bounds. The algorithms find matchings that are not too far away from the optimal (utilitarian) matching by employing certain strategies, *e.g.* dividing the plane into strips. Vaidya (1989) presents an $O(n^{2.5} \log n)$ algorithm for the Euclidean bipartite matching problem. This bound was further improved to $O(n^{2+\epsilon})$ by Agarwal *et al.* (1999) for the optimal matching problem, called *minimum-weight bipartite Euclidean matching* there. Geometric bottleneck matching is considered by Efrat *et al.* (2001) who present an algorithm in $O(n^{1.5} \log n)$ time. We refine this algorithm here in order to compute globally maximal matchings efficiently, extending the given procedure by one additional outer loop (see Section 6).

Local search and approximation

In a multiagent scenario, not all agents may have global information, and a coach may be not available. Therefore, in this case, an optimal solution to the matching or another problem can only be found by *local search*. Local search techniques have been studied intensively as an interesting research topic on its own (see *e.g.* Papadimitriou *et al.* (1998) and Jungnickel (2005)). The complexity of local search for the traveling salesperson problem (finding the shortest round-trip) is discussed *e.g.* in Papadimitriou *et al.* (1977). For this problem, *k-change search* is considered, where at most k edges of a tour are exchanged at a time. We adapt this concept for $k = 2$ in the decentralized matching algorithms considered here (Section 5). Papadimitriou (1977) shows however that there are instances of the traveling salesperson problem, where such a heuristic is ineffective. Similar examples can be found in our context, indicating that locally maximal matchings may be worse than globally maximal matchings (see Section 5.2) which can be found in $O(n^{2.5} \log n)$, *i.e.* polynomial time (Theorem 6.1). Nevertheless, a polynomial time approximation scheme for the NP-complete Euclidean traveling salesperson problem is known (Arora, 1998).

4. Local and global matchings

From a multiagent systems perspective, the problem with the most algorithms considered so far is the need of central control or negotiation among more than two agents. What happens if we allow only communication and exchange of partners among at most two agents and opponents in order to improve the matching? Another question is: are the properties considered so far really strong enough for the considered applications? — This leads us to the following definition, that gives us stronger versions of the properties from Definition 3.4 (global matchings), or allows us to restrict attention to submatchings only (pairs of agents in local matchings). We address these questions with the following (new) definitions.

DEFINITION 4.1 Let M be a matching between P and Q, and $\mathcal{P}$ be one of the properties from Definition 3.4. Then, we define:

1) M *globally* satisfies $\mathcal{P}$ iff $\mathcal{P}$ holds for all submatchings $M' \subseteq M$.

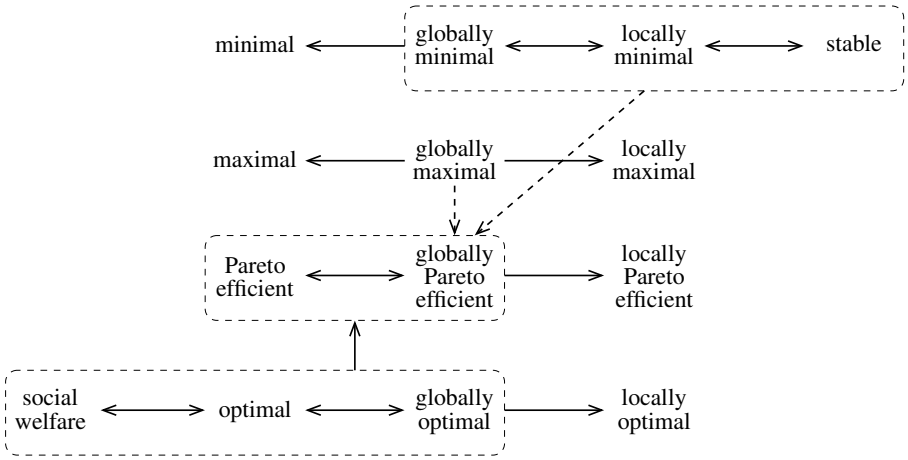


Figure 2. *Implication graph for matching properties. Solid double arrows indicate (bi)implications. For dashed arrows, the condition that all distances are pairwise different is additionally required. Equivalent concepts are comprised in dashed boxes*

2) M *locally* satisfies $\mathcal{P}$ iff $\mathcal{P}$ holds for all submatchings $M' \subseteq M$ with $\left|M'\right| = 2$, *i.e.* for all subsets of cardinality 2.

4.1. *Relationships among matching properties*

Clearly, global properties imply the simple properties from Definition 3.4 and their local versions. But local properties, in general, do not imply the respective (simple) property. For example, the matching drawn with solid lines in Figures 1c, where all distances are 3 units long, is only locally optimal, maximal, and Pareto efficient. Any pairwise exchange of two edges (indicated with dotted lines) does not improve the matching wrt. to these properties, because the length 5 would occur then. However, the matching drawn with dashed lines, where all distances are 2 units long, is (globally) optimal, maximal and Pareto efficient. Furthermore, minimal matchings are not always locally or globally minimal. To see this, consider Figure 1d after setting $d(W, w) < 1$. Figure 1d also shows that maximal and locally maximal matchings may be different. Interestingly, in the former example (Figure 1c) the globally maximal matching coincides with the maximal matching, whereas in the latter example (Figure 1d) it coincides with the locally maximal one. The following theorems state some relationships among the properties. Figure 2 summarizes the relationships among them.

THEOREM 4.2 A matching M is globally minimal iff it is locally minimal, *i.e.* stable.

Proof: The direction from left to right is trivial. Since M is globally minimal, every submatching $M' \subseteq M$ is a minimal matching, *i.e.* especially where $|M'| = 2$. Hence, M is locally minimal.

We prove the direction from right to left by showing its contraposition. Thus, let M be a matching between P and Q that is not globally minimal. Therefore, there exists a submatching $M_1 \subseteq M$ between some $P' \subseteq P$ and $Q' \subseteq Q$ and another matching M_2 between P' and Q' such that $d_0 = \min_{p \in P'} d_{M_2}(p) < \min_{p \in P'} d_{M_1}(p)$. Note that both M_1 and M_2 must contain at least two elements, because they must be different. Let now $\{p_i, q_j\} \in M_2$ with $d(p_i, q_j) = d_0$ and $M' = \{\{p_i, q_i\}, \{p_j, q_j\}\}$ be a two-element submatching of M_1 (with p_i and q_j as above). Then, $\min(d(p_i, q_j), d(p_j, q_i)) = d_0 < \min(d(p_i, q_i), d(p_j, q_j))$ by definition of d_0. Hence, M is not locally minimal, because M' (of cardinality 2) is not minimal. ■

THEOREM 4.3 A matching M is Pareto efficient iff it is globally Pareto efficient.

Proof: We prove the direction from left to right by showing its contraposition. Thus, let M be a matching between P and Q that is not globally Pareto efficient. Therefore, there exists a submatching $M_1 \subseteq M$ between some $P' \subseteq P$ and $Q' \subseteq Q$ that is not Pareto efficient, and another matching M_2 between P' and Q' which is Pareto efficient, hence $d_{M_2}(p_1) < d_{M_1}(p_1)$ for some $p_1 \in P'$ and $d_{M_2}(p_2) \leq d_{M_1}(p_2)$ for all $p_2 \in P'$. Let now $M' = (M \setminus M_1) \cup M_2$, *i.e.*, we extend M_2 to a complete matching between P and Q. Since $M_1 \subseteq M$, $M_2 \subseteq M'$, and $M \setminus M_1 = M' \setminus M_2$, *i.e.* $d_M(p) = d_{M'}(p)$ for all $p \in P \setminus P'$, it follows $d_{M'}(p_1) < d_M(p_1)$ for some $p_1 \in P' \subseteq P$ and $d_{M'}(p_2) \leq d_M(p_2)$ for all $p_2 \in P$. Hence, M is not Pareto efficient.

The direction from right to left is trivial. Since M is globally Pareto efficient, every submatching $M' \subseteq M$ is Pareto efficient, *i.e.* especially $M' = M$. Hence, M is Pareto efficient. ■

THEOREM 4.4 If a matching M is optimal, then it is also Pareto efficient, but not *vice versa*.

Proof: We address the theorem by proving its contraposition. Thus, if M is not Pareto efficient, there exists another matching M' such that $d_{M'}(p_1) < d_M(p_1)$ for some $p_1 \in P$ and $d_{M'}(p_2) \leq d_M(p_2)$ for all $p_2 \in P$. Hence, M cannot be optimal, because this implies immediately $\sum_p d_{M'}(p) < \sum_p d_M(p)$.

To see that the converse does not hold (*i.e.*, Pareto efficiency does not imply optimality), look at Figure 1a. The dashed solution is Pareto efficient, but not optimal. ■

THEOREM 4.5 A matching M is optimal, *i.e.* optimizes the social welfare, iff it is globally optimal.

Proof: We prove the direction from left to right indirectly. Thus, let M be an optimal matching between P and Q, but M_1 be a (proper) submatching of M between some

$P' \subseteq P$ and $Q' \subseteq Q$ that is not optimal. Clearly, there must be another matching M_2 between P' and Q' that is optimal. Then, we have $\sum_{p \in P'} d_{M_2}(p) < \sum_{p \in P'} d_{M_1}(p)$ by definition. Clearly, $M' = (M \setminus M_1) \cup M_2$ is a matching between P and Q. But it holds:

$$\begin{aligned}
\sum_{p \in P} d_{M'}(p) &= \sum_{p \in P \setminus P'} d_{M'}(p) + \sum_{p \in P'} d_{M'}(p) \\
&= \sum_{p \in P \setminus P'} d_{M}(p) + \sum_{p \in P'} d_{M_2}(p) \\
&< \sum_{p \in P \setminus P'} d_{M}(p) + \sum_{p \in P'} d_{M_1}(p) = \sum_{p \in P} d_{M}(p)
\end{aligned}$$

Hence, M cannot be optimal, which contradicts our assumption.

The direction from right to left is trivial. Since M is globally optimal, every submatching $M' \subseteq M$ is optimal, *i.e.* especially $M' = M$. Hence, M is optimal. ■

4.2. *Maximal matchings*

Maximal matchings can often be improved, as Figure 1d shows, because they are not even locally maximal in general. Furthermore, Figure 1f shows that (simple) maximality does not imply Pareto efficiency, yet it implies weak Pareto efficiency (Moulin, 1988, Lemma 1.1a). Figure 1c shows that local maximality does not imply Pareto efficiency, too. Only *global* maximality implies Pareto efficiency (see Theorem 4.6). Therefore, we concentrate on the problem of computing globally maximal matchings in the sequel. However, this implication is only valid if all possible distances between points in P and Q are different, look at Figure 1e.

THEOREM 4.6 Let $d(p, q)$ be pairwise different for all $p \in P$ and $q \in Q$. If M is a globally maximal matching between P and Q, then M is Pareto efficient.

Proof: We give the proof by showing its contraposition. Thus, if M is not Pareto efficient, there exists another matching M' such that $d_{M'}(p_1) < d_M(p_1)$ for some $p_1 \in P$ and $d_{M'}(p_2) \leq d_M(p_2)$ for all $p_2 \in P$. Let now P' be the set of all $p \in P$ satisfying $d_{M'}(p) \neq d_M(p)$, which means $d_{M'}(p) < d_M(p)$ in this context. In addition, we define $M_0 = \{\{p, q\} \mid p \in P',\ q \in Q\}$, $M_1 = M_0 \cap M$, and $M_2 = M_0 \cap M'$.

Since $d(p, q)$ are pairwise different for all $p \in P$ and $q \in Q$, it must be $M \setminus M_1 = M' \setminus M_2$. Therefore, M_1 and M_2 are matchings between P' and Q' for one and the same set $Q' \subseteq Q$. Since $d_{M_2}(p) < d_{M_1}(p)$ for all $p \in P'$, M_1 cannot be a maximal matching between P' and Q'. Hence, M cannot be a globally maximal matching between P and Q, because M_1 is a submatching of M. ■

A similar theorem holds for globally minimal matchings (see below). Clearly, a minimal matching is not necessarily globally minimal, *i.e.* stable. To see this, consider Figure 1f after setting $d(K, k) = 3$. Then, the dashed matching is minimal, but neither globally nor locally minimal.

THEOREM 4.7 Let $d(p, q)$ be pairwise different for all $p \in P$ and $q \in Q$. If M is a globally or locally minimal matching between P and Q (*i.e.*, M is stable), then M is Pareto efficient.

Interestingly, if all possible distances are pairwise different, then global maximality coincides with (the minimization wrt.) a special social welfare ordering, that we call *leximax*, which implies (simple) maximality (*i.e.*, it is egalitarian) and also Pareto efficiency even without this precondition (Moulin, 1988, Lemma 1.1b). For this, we have to consider the list L_M of distances $d(p, q)$ for all $(p, q) \in M$, sorted in decreasing order, *i.e.*, the maximal distances are in the beginning, and the *lexicographic ordering* $\ll$ for lists $L = \langle l_1, \ldots, l_n \rangle$ and $L' = \langle l'_1, \ldots, l'_n \rangle$ with $L \ll L'$ iff there exists an i (with $1 \leq i \leq n$) such that $l_1 = l'_1, \ldots, l_{i-1} = l'_{i-1}$, but $l_i < l'_i$. The goal is now to find a matching corresponding to a list of distances which is minimal wrt. $\ll$. In computer science and combinatorial optimization, this problem is also known as *lexicographic bottleneck matching* (Burkard *et al.*, 1991; Della Croce *et al.*, 1999). We now prove the above-mentioned coincidence (Theorem 4.9) and beforehand a useful technical lemma.

LEMMA 4.8 Let M be a matching between the sets P and Q that is not globally maximal. Then, there exists another matching M' between P and Q with $L_{M'} \ll L_M$. Furthermore, M' can be written as $(M \setminus M_1) \cup M_2$, for any not maximal matching M_1 and maximal matching M_2, which are submatchings of M and M', respectively, wrt. the same sets $P' \subseteq P$ and $Q' \subseteq Q$.

Proof: Since M is not globally maximal, there exists a submatching $M_1 \subseteq M$ between some $P' \subseteq P$ and $Q' \subseteq Q$ that is not maximal, and another matching M_2 between P' and Q' which is maximal. Let now $M' = (M \setminus M_1) \cup M_2$ as required. We further define $d_1 = \max_p d_{M_1}(p)$ and $d_2 = \max_p d_{M_2}(p)$. Clearly, $d_1 > d_2$.

Let i be the number of elements $p \in P$ with $d_M(p) \geq d_1$ and i' the respective number with $d_{M'}(p) \geq d_1$. Since $d_1 > d_2$ and both are the maximal distances occurring in $M_1 \subseteq M$ and $M_2 \subseteq M'$, respectively, and $M \setminus M_1 = M' \setminus M_2$, we have $i > i'$. It follows further that, for the distance lists $L_M = \langle l_1, \ldots, l_n \rangle$ and $L_{M'} = \langle l'_1, \ldots, l'_n \rangle$ (sorted in decreasing order), it holds $l_1 = l'_1, \ldots, l_{i'} = l'_{i'}$, but $l_{i'+1} = d_1 > l'_{i'+1}$. Hence, it follows $L_{M'} \ll L_M$. ■

THEOREM 4.9 If M is a lexicographic bottleneck matching between P and Q, then M is globally maximal. If $d(p, q)$ are pairwise different for all $p \in P$ and $q \in Q$, then the converse holds, too.

Proof: We prove the first claim by showing its contraposition. Thus, let M be a matching that is not globally maximal. By Lemma 4.8, there exists another matching M' with $L_{M'} \ll L_M$. Hence, M is not a lexicographic bottleneck matching.

We prove the second claim by showing the contraposition of the converse of the first claim. Thus, let M be a matching that is not a lexicographic bottleneck matching and M' be another matching between P and Q that has this property. For the corresponding lists of distances $L_M = \langle l_1, \ldots, l_n \rangle$ and $L_{M'} = \langle l'_1, \ldots, l'_n \rangle$ (sorted in decreasing order), it must hold $L_{M'} \ll L_M$. Let now i (with $1 \leq i \leq n$) be the smallest index with $l_i \neq l'_i$, which means $l_i > l'_i$ in this context. Consider now the submatchings $M_1 \subseteq M$ and $M_2 \subseteq M'$ that correspond to the distance lists $\langle l_i, \ldots, l_n \rangle$ and $\langle l'_i, \ldots, l'_n \rangle$, respectively. Since all distances $d(p, q)$ are pairwise different for all $p \in P$ and $q \in Q$ by precondition, M_1 and M_2 must be uniquely determined matchings wrt. the same sets $P' \subseteq P$ and $Q' \subseteq Q$. Because of $l_i > l'_i$, it follows that M_1 is not a maximal submatching of M. Hence, M is not globally maximal. ■

In order to see that the precondition in the second part of the last theorem is necessary, we consider the example in Figure 1e once again, where not all distances are pairwise different. Both solutions are globally maximal, but the dashed solution is not a lexicographic bottleneck matching. This shows that not every globally maximal matching is a lexicographic bottleneck matching.

5. Decentralized matching

In this section, we provide a decentralized algorithm *LocalMatch* for calculating local matchings. The algorithm is able to deal with partial matchings and requires only communication between pairs of agents. For this, we first have to reformulate the local matching definitions from Definition 4.1 for use in the pairwise algorithm.

DEFINITION 5.1 Let $M = \{\{p_1, q_1\}, \ldots, \{p_n, q_n\}\}$ be a matching between P and Q with pair $\{p_i, q_j\} \notin M$, but $\{p_i, q_i\} \in M$ and $\{p_j, q_j\} \in M$. Then, M is called

1) *locally minimal* wrt. $\{p_i, q_j\}$ iff $\min(d(p_i, q_j), d(p_j, q_i)) \geq \min(d(p_i, q_i), d(p_j, q_j))$,

2) *locally maximal* wrt. $\{p_i, q_j\}$ iff $\max(d(p_i, q_j), d(p_j, q_i)) \geq \max(d(p_i, q_i), d(p_j, q_j))$,

3) *locally Pareto efficient* wrt. $\{p_i, q_j\}$ iff $d(p_i, q_j) > d(p_i, q_i)$ or $d(p_j, q_i) > d(p_j, q_j)$, or $d(p_i, q_j) = d(p_i, q_i)$ and $d(p_j, q_i) = d(p_j, q_j)$, and

4) *locally optimal* wrt. $\{p_i, q_j\}$ iff $d(p_i, q_j) + d(p_j, q_i) \geq d(p_i, q_i) + d(p_j, q_j)$.

5.1. *A non-deterministic algorithm*

Figure 3 states a non-deterministic algorithm for local matching in pseudo-code. There, we write $\mathcal{P}(p_i, q_j)$ iff $\{p_i, q_j\} \notin M$ satisfies the respective property from

Definition 5.1, and $\mathcal{P}(M)$ means that $\mathcal{P}(p_i, q_j)$ holds for all such $\{p_i, q_j\}$. Further, a vertex $v \in P \cup Q$ incident with an edge in the (possibly partial) matching M is called *matched*. Otherwise, it is called *exposed*. Following the lines of Efrat *et al.* (2001), we define:

DEFINITION 5.2 (Paths) A *path* $\pi = \langle v_1, \ldots, v_k \rangle$ is a sequence of distinct positions in $P \cup Q$. It is identified with its set of edges $\{\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{k-1}, v_k\}\}$. π is called an *alternating path* iff for $1 \leq i < k$, we have $v_i \in P$ iff $v_{i+1} \in Q$, and, for any two consecutive edges $\{v_{j-1}, v_j\}$ and $\{v_j, v_{j+1}\}$ in π, for $1 < j < k$, it holds $\{v_{j-1}, v_j\} \in M$ iff $\{v_j, v_{j+1}\} \notin M$. π is called an *augmenting path* iff it is alternating and v_1 is an exposed vertex in P and v_k is an exposed vertex in Q. Finally, we define $M \oplus \pi = (M \backslash \pi) \cup (\pi \backslash M)$, *i.e.* as symmetric difference between the respective sets of edges.

Intuitively the algorithm works as follows: An agent $p_i \in P$ selects a partner $q_j \in Q$ if it is free (*i.e.* not matched). If p_i already has a partner q_i, but q_j is closer and not matched so far, then p_i will take q_j instead of q_i. If q_j is already matched and one of the properties $\mathcal{P}$ from Definition 5.1 is not satisfied, the relevant partners are swapped (in the procedure *Swap*). The algorithm exploits the following theorem:

THEOREM 5.3 A matching M is locally minimal, locally maximal, locally Pareto efficient, or locally optimal, respectively, (according to Definition 4.1) iff the respective property holds wrt. all pairs $\{p_i, q_j\} \notin M$ (according to Definition 5.1).

The procedure *LocalMatch* is a *multiagent algorithm* requiring communication among pairs of agents. Each agent (with position p) has only local information. It only needs to know the n distances $d(p, q_k)$ for $1 \leq k \leq n$, but not necessarily all distances for the other agents. Let $\{p_i, q_i\}$ and $\{p_j, q_j\}$ be in M. Now, agent p_i may offer agent p_j a swap of their partners, which should be accepted by p_j iff the respective property $\mathcal{P}$ will be satisfied after the swap, *i.e.*, the partners of p_i and p_j are successfully exchanged by the procedure call *Swap*(p_i, q_j) in this case. Note that in principle, two agents must communicate only the distances to their actually assigned partner and a feedback whether the offered *Swap* operation is acceptable, *i.e.* successful, or not.

LocalMatch is also a *non-deterministic algorithm* with respect to which pair is chosen in the procedure call *Swap*(p_i, q_j). This could be made completely at random, because the procedure always reaches a local matching after a finite number of *Swap* operations (see Theorem 5.4). Note that the overall multiagent algorithm terminates iff no further swap is possible for any of the agents. Nevertheless the procedure may be aborted at any time, yielding a possible matching, which might be suboptimal, but still useful. In any case, termination is guaranteed because of the following theorem:

THEOREM 5.4 (TERMINATION) The procedure *LocalMatch* from Figure 3 terminates for every property $\mathcal{P}$ from Definition 5.1.

```
funct LocalMatch(P, Q)
  begin
    M ← ∅
    while ¬P(M) ∨ Partial(M) do
          Swap(p_i, q_j) where p_i ∈ P, q_j ∈ Q
    od
    return(M)
  end

proc Swap(p_i, q_j)
  begin
    if {p_i, q_j} ∈ M then exit fi
    if Exposed(p_i)
      then if Exposed(q_j)
             then M ← M ⊕ ⟨p_i, q_j⟩
              else let {p_j, q_j} ∈ M
                   if d(p_i, q_j) < d(p_j, q_j) then M ← M ⊕ ⟨p_i, q_j, p_j⟩ fi
           fi
       else let {p_i, q_i} ∈ M
           if Exposed(q_j)
             then if d(p_i, q_j) < d(p_i, q_i)
                    then M ← M ⊕ ⟨q_j, p_i, q_i⟩ fi
              else let {p_j, q_j} ∈ M
                   if ¬P(p_i, q_j) then M ← M ⊕ ⟨p_i, q_j, p_j, q_i, p_i⟩ fi
           fi
    fi
  end
```

Figure 3. *Pseudo-code for algorithm* LocalMatch *for local matching*

Proof: We first consider the property local maximality. Let therefore M contain a two-element submatching $M_1 = \{\{p_i, q_i\}, \{p_j, q_j\}\}$ which is not maximal, hence M is not globally maximal. Let M' be the result of the (successful) application of the *Swap* procedure wrt. the elements in M_1. It follows $L_{M'} \ll L_M$ because of Lemma 4.8 where $M_2 = \{\{p_i, q_j\}, \{p_j, q_i\}\}$. Since the length of the lists is constantly n, and there are only finitely many different lengths l that are possible, $\ll$ is a strict well-ordering, *i.e.* a total order without any infinite descending chains. Hence, the procedure will eventually terminate.

For local minimality (*i.e.* stability), the proof is completely analogous. However, in this case, the distances $d(p, q)$ in L have to be sorted in increasing order. For Pareto efficiency termination is clear, because at least one distance $d_M(p)$ is decreased for

one $p \in P$ in each step, while the others remain the same. If $\mathcal{P}$ is local optimality, then simply the sum of lengths decreases monotonically. ■

Since (as just proven) termination is always guaranteed independently of the choice of *Swap* operations, termination of the overall procedure can also be detected from the perspective of an individual agent that currently cannot offer any swap, simply by watching the neighboring k agents for some time t (depending on k among others), whether they still perform swaps. Furthermore, the procedure can be speeded up by some heuristics. For instance, closer partners should be preferred, *i.e.*, an agent p should consider partners q in ascending order of the corresponding distances $d(p, q)$. In addition, if an agent p receives more than one swap offer at a time, it can choose the better one.

In the distributed algorithm *LocalMatch*, we only consider the agents pairwise. A whole suite of algorithms could be developed starting from the local pairwise matching approach to a completely centralized approach depending on the number of agents that are involved. This leads to the field of k-coordinated algorithms for distributed constraint optimization problems (Maheswaran *et al.*, 2004), where k agents coordinate to select their variable values.

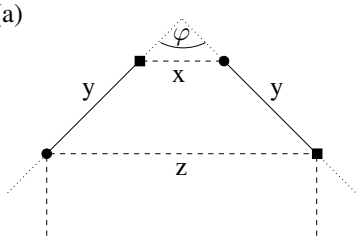
5.2. *Worst-case results for local matching*

We notice that there may be more than one (locally) maximal matching for given point sets P and Q. For example, in Figure 1c both solutions are locally maximal, and in Figure 1f both are maximal, whereas the globally maximal matching is uniquely determined in general (see Section 6). As we will see by the next examples, there are even more differences between locally and globally maximal matchings. First, locally maximal and even locally optimal matchings can be infinitely worse than (globally) maximal and optimal ones (see Example 1 below). And second, the run-time complexity of the *LocalMatch* procedure for locally maximal matching, *i.e.* the number of successful *Swap* operations (which may occur repeatedly) until a locally maximal matching is reached, may be quite high for special examples (see Example 2).

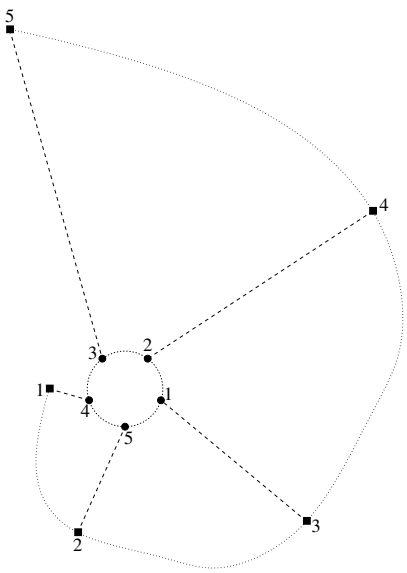
Example 1 (Regular n-gon) The first example is a generalization of Figure 1c, showing that lengths may be arbitrarily bad for local matchings. Consider a regular n-gon ($n \geq 3$) with unit edges (*i.e.* of length 1). The angle in each corner is then $\varphi = \pi(n-2)/n$ (with $\pi = 3.14159\ldots$ in this context). Now we add lines of length x in each corner cutting off the tips, the length y remains from each side, and z is as shown in Figure 4a. The $2n$ agents from P and Q are alternately placed at all corners. In Figure 4a, this is sketched for $n = 4$ where only one original corner is shown. Now, by the intercept theorems, we get:

$$\frac{x/2}{(1-y)/2} = \sin(\varphi/2) = \frac{z/2}{(1+y)/2} \qquad [1]$$

(a)



(b)



(c)

n	swaps	repeats
1	0	0
2	1	1
3	4	1
4	14	2
5	38	3
6	105	7
7	286	13
8	734	29

Figure 4. *Matching examples revealing the worst-case performance of locally maximal matching: (a) long distances in matchings (Example 1), (b) long swap sequences (Example 2), and (c) their lengths*

The solution with n times distance x is clearly a maximal and optimal matching. But the solution with n times distance y is locally maximal and locally optimal, if the two conditions $z \geq y \geq x$ and $z+x \geq 2y$ hold. Note that all other possible connecting lines are longer than y and hence need not be considered. If we take $z + x = 2y$, *i.e.* $y = x/2 + z/2$, which implies the two conditions from above, we obtain:

$$y = \sin(\varphi/2) \qquad [2]$$

To see this, we remove the denominators of both equations in [1] and get $x/2 = \sin(\varphi/2)(1-y)/2$ and $z/2 = \sin(\varphi/2)(1+y)/2$ which implies [2]. To see this, add both equations. It follows that locally maximal matchings can be infinitely worse than the (globally) maximal and optimal ones, because the ratio between x and y converges to zero:

$$x/y = 1 - y = 1 - \sin(\varphi/2) = 1 - \sin(\pi(1/2 - 1/n)) \quad \xrightarrow[n\to\infty]{} \quad 0$$

Example 2 (Spiral) Let us consider another example, showing that it may take long until a locally maximal matching is reached. Here, the agent positions in Q form a circle (strictly speaking, a regular n-gon as above), whereas the other agent positions in P form a spiral. This is shown in Figure 4b for $n = 5$. The polar coordinates of the positions in P and Q are given by $p_k = (2k, 2\pi(k - n \div 2 - \epsilon)/n)$ (where $\div$ means

integer division) and $q_k = (1, 2\pi\, k/n)$, respectively, for $1 \leq k \leq n$ and some small ϵ with $0 < \epsilon < 1/2$. Note that, for all i and j (with $1 \leq i, j \leq n$) and $k < k'$, it holds:

$$d(p_k, q_i) < d(p_{k'}, q_j)$$

The sequences of successful *Swap* operations starting with the initial matching $M_0 = \{\{p_1, q_1\}, \ldots, \{p_n, q_n\}\}$ and ending with a locally maximal matching M_1, which is uniquely determined in this case, may be rather long for this example. The dashed lines in Figure 4b indicate M_1, while M_0 is not shown. The maximal numbers of *Swap* operations, *i.e.* the numbers of subroutine calls, are listed in the table in Figure 4c. They have been computed by a small program (written by the first author in Prolog), implementing an exhaustive search. A successful *Swap* operation can be characterized by two pairs $\{p_i, q_i\}$ and $\{p_j, q_j\}$. There are altogether n^2 possibilities for choosing the first pair, and $(n-1)^2$ for the second one. In addition it is clear that, if a *Swap* operation is successful with the pairs $\{p_i, q_i\}$ and $\{p_j, q_j\}$, then this is not the case for the pairs $\{p_i, q_j\}$ and $\{p_j, q_i\}$. Hence, there are at most $n^2(n-1)^2/2$ different *Swap* operations possible for given sets P and Q. Interestingly however, the computed longest sequences contain one and the same *Swap* operation more than once. The last column of the table in Figure 4c lists the maximal numbers of repeating one and the same *Swap* operation. For $n = 8$ *e.g.*, the *Swap* operation with the pairs $\{p_7, q_8\}$ and $\{p_8, q_6\}$ occurs 29 times in the longest computed sequence with overall length 734.

Both examples show, that it is really worthwhile to consider global matching algorithms (see also Section 4.2). However, such cases will emerge very seldom in practice, and experiments with a prototypical implementation of the procedure *LocalMatch* for locally maximal matching show its good practical efficiency. In addition, the performance can simply be improved by just preferring closer partners if there is a choice among several partners as already mentioned in Section 5.1 and by avoiding repeated applications of one and the same *Swap* operation.

6. Globally maximal matching

Let us now come to the problem of computing a globally maximal matching. The problem of computing a maximal (or bottleneck) matching has been treated in the literature, see *e.g.* Efrat *et al.* (2001). But simple maximality often is not enough, as we have already seen in Section 4.2. We should take global maximality into consideration. In order to achieve this, we extended the algorithm for bottleneck matching given in Efrat *et al.* (2001). Figure 5 shows the new procedure for computing *globally* maximal matchings. It extends and refines the procedure in Efrat *et al.* (2001) for bottleneck matchings by one additional outer loop, namely the one in the function *GlobalMatch*. There, we make use of the list L of all pairs $p \in P$ and $q \in Q$, that is sorted by ascending distances $d(p, q)$. Obviously, L contains n^2 elements. With p_k, q_k, and d_k, we denote the respective components of the k-th element in L where k

ranges from 1 to n^2. Since the maximal distance d_m is uniquely determined in each iteration of the loop in *GlobalMatch*, there is always exactly one globally maximal matching for any given point sets P and Q.

The function *Bottleneck* with index m computes a maximal matching M' for the given point sets P and Q according to the procedure in Efrat *et al.* (2001), taking into account only the distances less or equal than d_m. This is done by a binary search for a complete matching with minimized maximal distance in M, exploiting the fact that a matching is complete iff there is no augmenting path (Papadimitriou *et al.*, 1998, Theorem 10.1). It returns the index $k \leq m$ of the edge with maximal length d_k in the maximal matching M' just computed. After that, we consider the submatchings without distances greater than d_k. In this context, we assume that all distances are pairwise different, and hence the maximal element is uniquely determined.

The function $\mathit{Augment}_k(\pi)$ returns an augmenting path π if possible, where all distances are smaller or equal than the given bound d_k, starting from the exposed vertices in P, by constructing a layered graph according to Efrat *et al.* (2001), see also Figure 6. Note that its good complexity can only be obtained by making use of an abstract data structure $\mathcal{D}_r(Q')$ for a set of objects $Q' \subseteq Q$ and a fixed length r, supporting the operations $\mathit{neighbor}(\mathcal{D}_r(Q'), p)$ (returns an element $q \in Q'$ whose distance from $p \in P$ is at most r) and $\mathit{delete}(\mathcal{D}_r(Q'), q)$ (deletes the object q from Q'). In the Euclidean planar case (*i.e.* in $\mathbb{R}^2$), the *Bottleneck* function can be implemented by making use of unit disks (see Efrat *et al.* (2001, Section 5)). By this, maximal matchings can be computed efficiently in $O(n^{1.5} \log n)$ time. Since *GlobalMatch* contains one additional loop, exploiting the complexity result just mentioned, we get the following theorem for computing globally maximal matchings:

THEOREM 6.1 The procedure *GlobalMatch* requires time $O(n^{2.5} \log n)$ for computing a globally maximal matching M between P and Q.

7. Non-geometric matching

In the application scenarios presented in Section 2, two assumptions have implicitly been made. The first is that the agents are homogeneous, *i.e.*, that they have the same abilities. Further, it is assumed that the quality of a matching depends on the (geometric) distances between elements of the sets to be matched. But especially in the RoboCup scenario these assumptions may not hold for several reasons. First, the simulation allows heterogenous agents that differ slightly in their abilities. This fact does not only influence the time an agent needs to reach its target position, but it also affects the choice of an appropriate target, *e.g.* it is not good to assign an opponent to mark that can be reached quickly, but cannot be outplayed by an agent. Second, there are situations in which the agents have to reach their target positions as fast as possible, even if the resulting matching does not meet the distance-based requirements, while in other contexts time is not an issue. For most factors that affect the

```
funct GlobalMatch(P, Q)
  begin
    M ← ∅
    m ← n²
    L ← SortedList_{p∈P, q∈Q}(d(p, q))
    for k ← 1 to n do
        % find maximal distance d_m = d(p_m, q_m)
        m ← Bottleneck_m(P, Q)
        P ← P\p_m
        Q ← Q\q_m
        M ← M ⊕ ⟨p_m, q_m⟩
    od
    return(M)
  end

funct Bottleneck_m(P, Q)
  begin
    i ← 1
    j ← m
    % binary search of minimal maximal distance
    repeat M' ← ∅
           k ← ⌊(i+j)/2⌋
           while Augment_k(π) do M' ← M' ⊕ π od
           if Partial(M') then i ← k + 1 else j ← k fi
    until i = j
    return(i)
  end
```

Figure 5. *Pseudo-code for globally maximal matching*

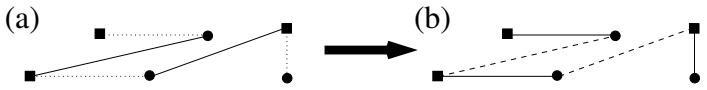


Figure 6. *Illustrating example for function* Augment$_k(\pi)$. *The solid lines connect the matched vertices. (a) shows an augmenting path starting with an exposed vertex from P and ending with another exposed vertex from Q. (b) shows the matching after performing the symmetric difference operation $\oplus$. In effect, the number of matched vertices is increased by one*

quality of a matching in the soccer application, deeper knowledge about the opponent agents is needed. While the behavior of the agents can only be modeled based on general knowledge about playing soccer, the individual abilities of an agent may be determined by observation.

As we already mentioned, in the RoboLog simulated soccer team the task of gaining information about opponent player types and calculating ranking-based matchings for man-to-man marking is fulfilled by the coach agent, which has the ability to determine opponent player types with an extraordinary reliability (Sturm, 2003). In the current implementation of the coach agent only distances between agents are used for calculating matchings among the players, but it is easy to integrate the additional knowledge gained by the coach into the ranking process.

8. Conclusion

In this paper we presented various types of distance-based matchings for situated multiagent systems. We defined different criteria for deciding the quality of such matchings and related them to well-known concepts in multiagent systems, *e.g.* Pareto efficiency. Several application scenarios of multiagent systems where decision making plays an important role have been given to motivate our work, including public transport, rescue scenarios, and robotic soccer. We described new algorithms for calculating local and global matchings in a decentralized and centralized way and provided proofs for the relationships among the different matching criteria.

As another example for centralized matchings calculated by a special agent, a so-called coach, we mentioned the online coach of the RoboLog soccer simulation team, where a ranking-based algorithm for computing stable matchings is already integrated in the man-to-man marking procedure (Sturm, 2003). With this implementation, players are assigned to opponents they have to mark during a free kick or kick in. Test games indicated that the team performance (measured by goal difference) can be slightly improved by this.

We can even go one step further. Sets of players from both teams can be selected based on their relevance for the current situation (*e.g.* distance to the ball and position on the field) and their player types (see Section 7). With those sets, a stable matching between teammates and opponents can be calculated. The players can then be told which opponent to mark based on this ranking. In the future, we will also examine the relationship of the presented methods for matching and certain kinds of contract nets (which are also decision systems), namely those which allow a *swap-operator* for agents (see Sandholm, 1999).

Acknowledgements

We would like to thank Dennis Farr and Jim Nastos for helpful responses to a question the first author asked in the usenet newsgroup comp.theory and Hitesh Saraf for the implementation of all matching algorithms, which are introduced in this paper, in Java. We also have to thank several anonymous referees, who read our paper very carefully and provided us with helpful comments and suggestions for considerably improving the original version. This research is partially supported by the grants *Fu 263/8* and *Sto 421/2* from the German research foundation *DFG* within the special priority program 1125 on *Cooperating Teams of Mobile Robots in Dynamic Environments*. A short and preliminary version of this paper appeared as Stolzenburg *et al.* (2003).

9. Bibliography

Agarwal P. K., Efrat A., Sharir M., « Vertical Decomposition of Shallow Levels in 3-Dimensional Arrangements and Its Applications », *SIAM Journal on Computing*, Vol. 29, No. 3, p. 912-953, 1999.

Arora S., « Polynomial Time Approximation Schemes for Euclidean TSP and other Geometric Problems », *Journal of the ACM*, Vol. 45, No. 5, p. 753-782, 1998.

Avis D., « A Survey of Heuristics for the Weighted Matching Problem », *Networks*, Vol. 13, p. 475-493, 1983.

Burkard R. E., Rendl F., « Lexicographic bottleneck problems », *Operations Research Letters*, Vol. 10, p. 303-308, 1991.

Chen M., Dorer K., Foroughi E., Heintz F., Huang Z., Kapetanakis S., Kostiadis K., Kummeneje J., Murray J., Noda I., Obst O., Riley P., Steffens T., Wang Y., Yin X., *RoboCup Soccer Server*. 2003, Manual for Soccer Server Version 7.07 and later (obtainable from `sserver.sf.net`).

Della Croce F., Paschos V. T., Tsoukiàs A., « An improved general procedure for lexicographic bottleneck problems », *Operations Research Letters*, Vol. 24, p. 187-194, 1999.

Efrat A., Itai A., Katz M. J., « Geometry Helps in Bottleneck Matching and Related Problems », *Algorithmica*, Vol. 31, p. 1-28, 2001.

Gale D., Shapely L., « College admissions and the stability of marriage », *American Mathematical Monthly*, Vol. 69, p. 9-15, 1962.

Irving R. W., Leather P., Gusfield D., « An Efficient Algorithm for the "Optimal" Stable Marriage », *Journal of the ACM*, Vol. 34, No. 3, p. 532-543, 1987.

Jungnickel D., *Graphs, Networks and Algorithms*, 2nd edn, Springer, Berlin, Heidelberg, New York, 2005.

Kok J. R., Spaan M. T. J., Vlassis N., « Multi-robot decision making using coordination graphs », A. de Almeida, U. Nunes (eds), *Proceedings of the 11th International Conference on Advanced Robotics, ICAR'03*, Coimbra, Portugal, p. 1124-1129, June, 2003.

Kuhlmann G., Stone P., Lallinger J., « The UT Austin Villa 2003 Champion Simulator Coach: A Machine Learning Approach », D. Nardi, M. Riedmiller, C. Sammut (eds), *RoboCup-2004: Robot Soccer World Cup VIII*, Springer Verlag, Berlin, 2005.

Kuhn H. W., « The Hungarian Method for the Assignment Problem », *Naval Research Logistics Quarterly*, Vol. 2, p. 83-97, 1955.

Maheswaran R. T., Pearce J. P., Tambe M., « Distributed Algorithms for DCOP: A Graphical-Game-Based Approach », *Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems*, San Francisco, CA, 2004.

Moulin H., *Axioms of cooperative decsion making*, Econometric Society Monographs No. 15, Cambridge University Press, Cambridge, 1988.

Murray J., Obst O., Stolzenburg F., « RoboLog Koblenz 2001 », A. Birk, S. Coradeschi, S. Tadokoro (eds), *RoboCup 2001: Robot Soccer World Cup V*, LNAI 2377, Springer, Berlin, Heidelberg, New York, p. 526-530, 2002. Team description.

Papadimitriou C. H., « The Euclidean Traveling Salesman Problem is NP-Complete », *Theoretical Computer Science*, Vol. 4, p. 237-244, 1977.

Papadimitriou C. H., Steiglitz K., « On the Complexity of Local Search for the Traveling Salesman Problem », *SIAM Journal on Computing*, Vol. 6, No. 1, p. 76-83, 1977.

Papadimitriou C. H., Steiglitz K., *Combinatorial Optimization: Algorithms and Complexity*, Dover edn, Dover Publications Inc., Mineola, NY, 1998.

Paquet S., Bernier N., Brahim C., « DAMAS-Rescue Description Paper », *Proceedings of the RoboCup Symposium 2004*, Lisbon, Portugal, 2004. Team Description Paper.

Reis L. P., Lau N., « FC Portugal 2004 Rescue Team Description: Adapting Simulated Soccer Coordination Methodologies to the Search and Rescue Domain », *Proceedings of the RoboCup Symposium 2004*, Lisbon, Portugal, 2004a. Team Description Paper.

Reis L. P., Lau N., Sampaio R., Marques H., Ferreira R., Penedones H., « FC Portugal Coach 2004: High-Level Coaching of a Heterogeneous Team using a Low-Level Language », *Proceedings of the RoboCup Symposium 2004*, Lisbon, Portugal, 2004b. Team Description Paper.

Riley P., Veloso M., Kaminka G., « An empirical study of coaching », H. Asama, T. Arai, T. Fukuda, T. Hasegawa (eds), *Proceedings of 7th International Symposium on Distributed Autonomous Robotic Systems*, Springer, Berlin, Heidelberg, New York, p. 215-224, 2002.

Sandholm T. W., « Distributed Rational Decision Making », G. Weiss (ed.), *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence*, MIT Press, Cambridge, MA, London, chapter 5, p. 201-258, 1999.

Stolzenburg F., Murray J., Sturm K., « Multiagent Matching Algorithms With and Without Coach », M. Schillo, M. Klusch, J. Müller, H. Tianfield (eds), *Proceedings of the 1st German Conference on Multiagent System Technologies*, LNAI 2831, Springer, Berlin, Heidelberg, New York, Erfurt, p. 192-204, 2003.

Sturm K., Der RoboLog-Coach – Ein Online-Coach für Fußballmannschaften der Simulationsliga der RoboCup-Initiative, Diplomarbeit No. D 690, Fachbereich Informatik, Universität Koblenz-Landau, 2003.

Vaidya P. M., « Geometry Helps in Matching », *SIAM Journal on Computing*, Vol. 18, No. 6, p. 1201-1225, 1989.

Visser U., Drücker C., Hübner S., E. S., Weland H.-G., « Recognizing Formations in Opponent Teams », *RoboCup 2000: Robot Soccer World Cup IV*, Lecture Notes in Artificial Intelligence, Springer, p. 391-396, 2001.