Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence

Dynamic Multi-Agent Task Allocation with Spatial and Temporal Constraints

Sofia Amador and **Steven Okamoto** and **Roie Zivan**
Industrial Engineering and Management Department
Ben-Gurion University of the Negev
Beer-Sheva, Israel
{amador, okamotos, zivanr}@post.bgu.ac.il

Abstract

Realistic multi-agent team applications often feature dynamic environments with soft deadlines that penalize late execution of tasks. This puts a premium on quickly allocating tasks to agents, but finding the optimal allocation is NP-hard due to temporal and spatial constraints that require tasks to be executed sequentially by agents.

We propose FMC_TA, a novel task allocation algorithm that allows tasks to be easily sequenced to yield high-quality solutions. FMC_TA first finds allocations that are fair (envy-free), balancing the load and sharing important tasks between agents, and efficient (Pareto optimal) in a simplified version of the problem. It computes such allocations in polynomial or pseudo-polynomial time (centrally or distributedly, respectively) using a Fisher market with agents as buyers and tasks as goods. It then heuristically schedules the allocations, taking into account inter-agent constraints on shared tasks.

We empirically compare our algorithm to state-of-the-art incomplete methods, both centralized and distributed, on law enforcement problems inspired by real police logs. The results show a clear advantage for FMC_TA both in total utility and in other measures commonly used by law enforcement authorities.

Introduction

In law enforcement, police officers conduct routine patrols and respond to reported incidents (*tasks*). Each task has an *importance* ranging from low (e.g., noise complaint) to high (e.g., murder) and a *workload* indicating the amount of work that must be completed for the incident to be processed. Multiple officers may work together on especially important tasks, sharing the workload and improving response quality. Delays in arriving to the scene of an incident allow perpetrators to escape and situations to escalate. This is reflected by a *soft deadline* that decreases the utility derived from performing a task the later it is started. A task cannot be started until all assigned officers arrive at the location of the task, and this depends both on the relative locations of officers and tasks as well as on the sequence of other tasks that the officers are scheduled to perform. Finally, because new incidents may be reported at any time, allocations must react to dynamic changes with officers sometimes interrupting execution of low-priority tasks, even though this negatively impacts the performance of the interrupted task. We term the problem of finding allocations that maximize team utility over time the *Law Enforcement Problem (LEP)*.

In this paper we make three major contributions. First, we formalize LEP, which, despite its name, is also applicable to realistic multi-agent applications such as disaster response, multi-robot, and military teams. These domains share four properties that make task allocation extremely challenging: task heterogeneity, cooperative task execution, spatial and temporal constraints, and a dynamic environment. This combination of features has not previously been studied. Cooperative task execution imposes inter-agent constraints that increase the utility for shared tasks while decreasing the time a single agent must spend on the task; this comes at the expense of delaying execution until all agents arrive. Utility also depends on the order in which tasks are executed due to spatial-temporal constraints. Although LEP is NP-hard (the traveling salesman problem is a special case with a single agent), dynamism and soft deadlines require that solutions be found quickly before the environment changes or utility is lost due to agents' late arrivals to tasks.

Our second contribution is FMC_TA, a novel heuristic that requires worst-case polynomial and pseudo-polynomial time in centralized and distributed settings, respectively. Instead of directly searching the space of execution schedules, FMC_TA first finds an allocation by using a simplified problem model that ignores spatial-temporal and inter-agent constraints. Next, it schedules each agents' tasks based on the full problem model. The key challenge is to choose an allocation that can be scheduled to yield a high-quality solution.

FMC_TA borrows two concepts from non-cooperative multi-agent systems, *envy-freeness* and *Pareto optimality*, to do this. An allocation is envy-free if no agent prefers the allocation of another agent to its own, and is Pareto optimal if the team utility (i.e., *social welfare*) cannot be increased without any agent suffering a decrease in individual utility. Although agents in LEP are cooperative, we hypothesize that envy-freeness avoids long delays by balancing the task load among agents, while Pareto optimality helps to allocate high importance tasks to sets of agents with high capability. Thus, allocations with both properties will be able to be sequenced to yield high-quality solutions.

An envy-free, Pareto optimal allocation can be found by

using the Fisher market clearing (FMC) model. FMC_TA computes an FMC allocation by modeling agents as buyers, tasks as goods, and simplified utilities as preferences. Individual tasks, especially important ones, may be divided between multiple agents who share the workload.

FMC_TA then schedules the allocation, taking into account inter-task and inter-agent constraints. The tasks allocated to each agent are ordered by importance. These sequences are then modified based on inter-agent constraints: because shared tasks cannot be started until all assigned agents have arrived, agents do not benefit from arriving earlier than the latest agent. Hence, for each agent that is scheduled to arrive early to a shared task, we move its next scheduled task before the shared task if this does not delay the execution of the shared task.

Our third contribution is an empirical evaluation of FMC_TA, alternative task allocation algorithms, and general optimization algorithms in both centralized and distributed settings. We evaluated the algorithms on realistic problems based on actual log files provided by the police force in our home city. FMC_TA achieves higher team utility and also surpasses competitors on other measurements used by real-world law enforcement authorities to evaluate performance.

Related Work

Market-based task allocation (Jones, Dias, and Stentz 2007) has been used for dynamically allocating spatially-located tasks. Each task has an auctioneer, and agents compute the change in utility for being allocated a task, using either an optimal or near-optimal ordering, and submit this change as a bid. The auctioneer greedily and irrevocably allocates the task to the agent with the highest bid; task sharing is not permitted. This algorithm can be implemented centrally or distributedly, with different auctioneers for different tasks. Another market-based approach uses negotiation to allocate tasks with hierarchical structures mimicking product supply chains; however, spatial constraints are not considered and termination is not guaranteed (Walsh and Wellman 1998).

In the coalition formation with spatial and temporal constraints problem (CSFTP), agents form coalitions to jointly work on tasks with spatial constraints and deadlines. Agents sharing a task in CSFTP work at non-additive rates, and the objective is to maximize the number of tasks completed before hard deadlines. CFSTP was originally solved using Coalition Formation with Look-Ahead (CFLA) (Ramchurn et al. 2010a) that applies two heuristics: allocating the smallest possible coalition and maximizing the number of other tasks that can be completed before deadline in the next time step. CFLA's heuristics are not helpful when tasks can be performed by individuals (albeit with lower quality than with groups), deadlines are soft, and tasks have different inherent importance. CSFTP was later solved with the distributed max-sum DCOP algorithm (Ramchurn et al. 2010b; Macarthur et al. 2011), but the soft deadlines in LEP result in constraints of high arity that Max-sum requires exponential time to solve (Farinelli et al. 2008).

General optimization metaheuristics such as simulated annealing (Reeves 1993) and hill climbing with random restarts (Poole and Mackworth 2010) have also been used for task allocation. We found simulated annealing to dominate other metaheuristics for LEP, consistent with previous task allocation studies (Schoneveld, de Ronde, and Sloot 1997).

Market clearing models have been studied in economics for more than a century, starting with the work of Fisher and Walras from the end of the 19th century. Later work by Eisenberg and Gale (Gale 1960) proposed a convex program for solving the Fisher model. The Arrow-Debreu model, in contrast to Walras's preliminary work, is a market exchange model that guarantees the existence of a solution under some assumptions. More recent work has developed a polynomial-time centralized algorithm (Devanur et al. 2002) and a pseudo-polynomial time distributed algorithm (Zhang 2011). Our work is, to the best of our knowledge, the first attempt to apply a market clearing approach to realistic dynamic task allocation problems.

Law Enforcement Problem

In formalizing LEP we first consider the simpler static problem before turning to the full, dynamic version.

Static Problem

In the static LEP there are n cooperative, homogeneous agents (police units), $a_1, \ldots, a_n \in A$ and m tasks $v_1, \ldots, v_m \in V$ situated in a city, with the set of all possible locations denoted by L. The time it takes to travel between two locations is given by the function $\rho : L \times L \to [0, +\infty)$. With slight abuse of notation we write $\rho(a, v)$ or $\rho(v, v')$ to denote the travel times between locations of an agent and a task or between locations of two tasks, respectively.

There are two kinds of tasks: *patrols* of neighborhoods and *events* that require a police response. Each task v has an *importance* $I(v) > 0$; patrols are generally less important than events. Events also have a *workload* $w(v)$ specifying how much work (in time units, e.g., securing a crime scene, taking statements) must be performed to complete the task. Patrols have no workload but are ongoing tasks that are never completed. Agents derive positive utility at a constant rate while patrolling.

Multiple agents can share a single event (e.g., officers interviewing different witnesses). All agents sharing a task must be present before task execution can begin, and the amount of time an agent must spend on the task is equal to its allocated fraction of the workload; these portions may be of different sizes. The team utility of q agents working simultaneously on task v is denoted by the non-negative *capability* function, $Cap(v, q)$. This can represent minimum required or maximum allowed numbers of agents (by setting capability to 0 for fewer or more agents, respectively), as well as changes in execution quality due to synergies or coordination costs.

An allocation of tasks to agents is denoted by the $n \times m$ matrix X where entry x_{ij} is the fraction of task v_j that is assigned to a_i. Agents can only perform a single task at a time so the tasks allocated to agent a_i must be ordered into a schedule $\sigma^i = (v_{s_1^i}, t_1^i, t_1^{i\prime}), \ldots, (v_{s_{M_i}^i}, t_{M_i}^i, t_{M_i}^{i\prime})$ which is a sequence of M_i triples of the form $(v_{s_k^i}, t_k^i, t_k^{i\prime})$, where $v_{s_k^i}$ is the task performed from time t_k^i to $t_k^{i\prime}$. The spatial-temporal

constraints require that the time spent on each task must equal a_i's assigned share of the workload and that agents must have sufficient time to move between tasks because they can only perform tasks at their current location:

$$t^{i\prime}_k - t^i_k = x_{is^i_k} w(v_{s^i_k}) \qquad 1 \le k \le M_i \qquad (1)$$

$$t_{k+1} - t'_k \ge \rho(v_{s^i_k}, v_{s^i_{k+1}}) \qquad 1 \le k \le M_i \qquad (2)$$

Inter-agent constraints from task sharing require that all agents start a task at the same time: for all $v \in V$ there exists τ_v such that for all $a_i \in A$, $v = v_{s^i_{k_i}} \implies \tau_v = t_{k_i}$.

The utility of performing task v starting at time t depends on the capability of the agents performing the task and the *soft deadline* function $\delta(v,t) : V \times [0,+\infty) \to (0,1]$, which is monotonically non-increasing in t. Using the delay until a task is started is informed by real-world law enforcement settings, where conditions tend to worsen (e.g.,. perpetrators escape, confrontations escalate) until police officers arrive. In consultation with the police, we use an exponentially-decaying soft deadline function for events, $\delta(v,t) = \beta^{\gamma t}$, where $\beta \in (0,1]$ and $\gamma \ge 0$ are constants. For a patrol v there is no deadline so $\delta(v,t) = 1$.

Let q_v be the number of agents assigned non-zero portions of v, and sort those portions in non-decreasing order $y_{v1} \le y_{v2} \le \cdots \le y_{vq_v}$. Then after time $\tau_v + y_{vk} \cdot w(v)$ there are only $q_v - k$ agents still working on v. Thus the total utility of completing v is

$$U(v) = \delta(v,\tau_v) \sum_{k=1}^{q_v} y_{vk} Cap(v, q_v - k + 1)$$

and the total team utility is $U(V) = \sum_{v \in V} U(v)$.

Dynamic Problem

In the dynamic problem, tasks arise over time. We denote the *arrival time* of a task v by $\alpha(v)$. Tasks can only be assigned after arrival. The soft deadline decreases the value of tasks after their arrival, so $\delta(v,t_v) = \beta^{\gamma(t_v - \alpha(v))}$.

Because task arrival is unpredictable, the dynamic problem is represented as a sequence of static problems each instantiated when a new task arrives. When a new task arrives, the *current task* (if any) being performed by a_i is denoted CT_i. Agents can *interrupt* the performance of their current task. For example, an officer handling a (low importance) loitering complaint when a (high importance) murder is reported may be ordered to stop what he is doing and attend to the murder. Agents do not return to interrupted tasks.

Task interruption incurs a *penalty*, $\pi(v_j, \Delta w)$, which depends on the task v_j and the amount of work Δw completed when the task is interrupted. In consultation with the police, we assume that the penalty for an event v_j decreases exponentially with Δw to a minimum value:

$$\pi(v_j, \Delta w) = \max\{I(v_j) c^{w(v_j) - \Delta w}, \phi \cdot I(v_j)\},$$

where $c \in [0,1)$ and $\phi > 0$ are constants and $\phi I(v_j)$ is the minimum penalty. This is consistent with real police settings where the first few minutes are commonly critical and costly to interrupt. There is no penalty for interrupting a patrol, so $\pi(v_j, \Delta w) = 0$ in that case.

Letting $U'(v) = \delta(v,\tau_v) \sum_{k=1}^{q_v} Cap(v, q_v - k + 1) y_{vk}$, the utility of performing v in the dynamic problem is thus

$$U(v) = U'(v) - \sum_{a_i : v_{s_1} \ne CT_i} \pi(CT_i, \Delta w)$$

FMC-Based Task Allocation

A Fisher market contains n buyers, each endowed with an amount of money, and m goods. An $n \times m$ matrix R represents the preferences of buyers over products. A market-clearing solution is a price vector p specifying a price p_j for each good j that allows each buyer i to spend all her money on goods that maximize bang-per-buck (r_{ij}/p_j) while all goods in the market are sold. An FMC allocation is an $n \times m$ matrix X where each entry $0 \le x_{ij} \le 1$ is the fraction of good j allocated to buyer i given the market-clearing prices p. FMC allocations are Pareto optimal and also envy-free when monetary endowments are equal (Reijnierse and Potters 1998).

Centralized Algorithm

FMC_TA represents agents and tasks as buyers and goods, respectively, and endows each agent with an equal amount of money.[1] R is constructed by optimistically ignoring the inter-task ordering constraints and assuming the maximum value for the capability function. Specifically, we set entry r_{ij} at time t to be the utility of a_i immediately moving to v_j and performing it with the optimal number of other agents:

$$r_{ij} = \delta(v_j, t + \rho(a_i, v_j)) \max_q \{Cap(v_j, q)\} - \pi(CT_i, \Delta w) \qquad (3)$$

where the penalty is omitted if $CT_i = v_j$.

We find market-clearing prices using the polynomial-time algorithm of Devanur et al. (2002), then produce the allocation matrix X as described by Reijnierse and Potters (1998).

It is clear that R is not expressive enough to represent the complex team utility function in the formal LEP definition. However, in the simplified problem represented by R, the properties of the FMC allocation ensure that we achieve an efficient allocation that is balanced over the agents. Our experiments demonstrate that this results in higher team utility than directly maximizing the utility represented by R.

In the second stage of FMC_TA we schedule the allocated tasks for each agent to reflect the spatio-temporal inter-task and inter-agent constraints. We construct an ordering of tasks allocated to each agent a_i by greedily prioritizing tasks with higher maximum capability, as these will tend to lead to higher utility. Ties are broken in favor of older tasks, imposing a unique ordering across agents that prevents deadlock while also reducing task starvation, an important concern in law enforcement. Once the initial order is selected, we compute the initial schedule σ_i by setting t^i_k and $t^{i\prime}_k$ to satisfy Equations (1) and (2) at equality. This takes $O(m \log m)$ time to sort tasks and compute initial schedules.

We then update the start times to reflect the inter-agent constraints that shared task execution requires all agents to

[1] The use of money is purely an internal mechanism of the allocation algorithm.

be present. This delays shared tasks and any subsequent tasks, as reflected by the equations:

$$\tau_j = \max\{t_k^i \mid s_k^i = j \wedge x_{ij} > 0\} \quad (4)$$

$$t_k^i = \max\{t_{k-1}^i + \rho(v_{s_{k-1}^i}, v_{s_k^i}), \tau_{s_k^i}\}, \quad (5)$$

which can be solved by monotonically iterating through the sorted times in the initial schedules in $O(m \log m)$ time.

We next see if the order of tasks in the individual schedules can be optimized by moving tasks delayed by shared tasks earlier in the order without further delaying shared tasks. For a shared task $v_{s_k^i}$ with $k > 1$ and non-shared task $v_{s_{k+1}^i}$ with $k > 1$, the non-shared task is moved before the shared task if $\tau_{s_{k-1}^i} + \rho(v_{s_{k-1}^i}, v_{s_{k+1}^i}) + x_{is_{k+1}^i} w(v_{s_{k+1}^i}) + \rho(v_{s_{k+1}^i}, v_{s_k^i}) \leq \tau_{s_k^i}$. If $k = 1$ (the shared task is first) at time t, the non-shared task is moved if $t + x_{is_2^i} w(v_{s_2^i}) + 2\rho(a_i, v_{s_2^i}) \leq \tau_{s_1^i}$. If the schedules are changed, the start times are again updated using Equations (4) and (5). Each task is considered once, requiring $O(m)$ time.

FMC_TA thus runs in worst-case polynomial time to compute FMC allocations and schedule tasks centrally.

Distributed Algorithm

In the distributed setting each agent knows only about tasks within a certain *threshold* distance D and its own preferences for those tasks. We form a Fisher market where r_{ij} is set according to Equation (3) if $\rho(a_i, v_j) \leq D$ and 0 otherwise. Knowledge of R is distributed among the agents, each of whom only knows its non-zero entries. For each task there is a *seller* who computes the allocation for that task.

To compute a market clearing solution we use the distributed proportional response algorithm (Zhang 2011). Agents iteratively submit bids to sellers of the tasks within their threshold and are in turn awarded provisional allocations, which they use to modify their bids in the next round. This process converges in pseudo-polynomial time to an ε-approximate market-clearing solution and allocation.

Agents compute their initial schedules in the same way as in the centralized approach, then update start times for shared tasks. For each task $v_j = v_{s_k^i}$ allocated to it, agent a_i maintains an estimated start time τ_j^i, initialized to t_k^i. If v_j is a shared task, agent a_i then communicates (τ_j^i, j) to all other agents $a_{i'}$ who are sharing the task.

Upon receiving a (τ, j) message (where $j = s_k^i$), a_i computes the change in time $\theta = \tau - \tau_j^i$. If $\theta > 0$ then for all $k' \geq k$, a_i adds θ to $t_{k'}^i$ and $\tau_{k'}^i$, and transmits the new values $(\tau_{k'}^i, s_{k'}^i)$ to all agents sharing $v_{s_{k'}^i}$.

This process terminates in polynomial time with $\tau_v = \tau_v^i = \tau_v^{i'}$ for all $a_i, a_{i'} \in A, v \in V$ and Equations (4) and (5) satisfied. Because there are no deadlocks, at least one time t_k^i will become permanently fixed in each cycle, and so the number of communication cycles is bounded by the total number of tasks in the schedules. Each schedule can contain at most m tasks and there are n schedules, so the number of communication cycles is bounded by mn.

Running time of distributed FMC_TA is dominated by the pseudo-polynomial time to compute the FMC allocation.

Experimental Evaluation

We compare FMC_TA to state-of-the-art incomplete algorithms, both centralized and distributed. These approaches consider only discrete allocations. Thus, we selected in advance the maximum number q of agents that can share a task and considered only allocations in which $x_{ij} = z/q$, where $z \in \{0, 1, \ldots, q\}$. For comparison we applied these restrictions to FMC_TA by rounding the allocation matrix X.

The algorithms were tested on independently generated random problems based on real police logs; all results presented are averages of 20 simulated shifts.

The city is represented by a rectangular region of the Euclidean plane of size 6×6 kilometers, divided into 9 neighborhoods of size 2×2, each with a patrol task. We simulated 8-hour shifts as in real police departments, with 9 agents patrolling (one in each neighborhood) at the beginning of each shift. The number of tasks arriving (i.e., the *load*) could vary between shifts. Tasks arrived at a fixed rate and were distributed uniformly at random in the city.

There were four types of events of decreasing importance $I(v) = 2400, 1600, 1200, 800$ from type 1 to type 4, respectively. Patrols had $I(v) = 500$. Event types were selected randomly according to the distribution of real event types provided by law enforcement authorities in our home city: $30\%, 40\%, 15\%, 15\%$ of events were of type 1 to 4, respectively. Based on estimates provided by the police, the workloads were drawn from exponential distributions with means $58, 55, 45, 37$ for events of type 1 to 4, respectively.

We assumed Cap improved quality of task execution for each agent up to a maximum number of agents Q_v:

$$Cap(v, q) = \min\{\frac{q}{Q_v} I(v), I(v)\}$$

where $Q_v = 3, 2, 1, 1$ for tasks of type 1 to 4, respectively. The discount function used was $\delta(v, t) = 0.9^t$ for all v.

Evaluation of Centralized Algorithms

We compared centralized FMC_TA to three state-of-the-art incomplete algorithms. Simulated annealing (SA) (Reeves 1993) optimized allocations that were scheduled using the same heuristic as FMC_TA. SA was rerun after each dynamic event, using the previous allocation as the starting allocation for the subsequent run. Market-based task allocation (MBTA) (Jones, Dias, and Stentz 2007) used exponential-time complete search to optimal schedules. CFLA$^+$ is a version of the CFLA task allocation algorithm (Ramchurn et al. 2010a) adapted for LEPs. In the look-ahead phase, CFLA$^+$ computes the maximum utility for pairs of tasks taking into account soft deadlines.

We also tested a baseline approach identical to FMC_TA except that it calculates allocations using a linear program (LP) that directly maximizes the utility represented by R. Differences in solution quality between FMC_TA and LP thus reflect the effects of envy-freeness and Pareto optimality in the simplified problem. Note that LP does not share tasks because its constraint matrix is totally unimodular.

Figure 1 presents the utility achieved over time for loads of 60 events. Agents accumulate utility at a steady rate using all algorithms and a partial F-test confirmed that agents

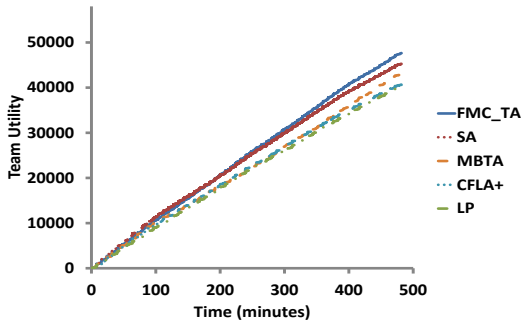

Figure 1: Accumulated team utility, 60 events per shift.

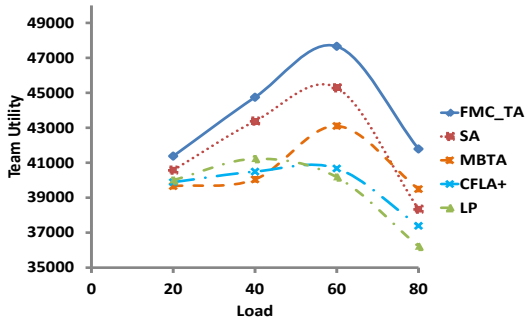

Figure 2: Accumulated team utility for varying loads.

using FMC_TA accrue utility significantly faster than with the other approaches. LP does worst of all, confirming that directly trying to maximize utility in the simplified problem is insufficient for achieving high utility in the full problem.

FMC_TA also dominates the other approaches for loads varying from 20 to 80, as seen in Figure 2. SA generally does well as expected of a metaheuristic directly optimizing the team utility objective. Although they are specialized task allocation algorithms, CFLA$^+$ and MBTA were not designed for the challenging combination of properties of LEP and hence do less well. MBTA does poorly for low loads, possibly because it does not share tasks; it was designed for oversubscribed domains where the benefits of quickly reaching any task outweigh possible benefits of cooperation.

Team utility decreases for all algorithms when the load goes from 60 to 80. In FMC_TA, SA, and LP, this is primarily due to using task age in the scheduling heuristic to avoid starvation. While highly important to police, it means that under very high loads the system can become backlogged, with greater delay before tasks begin execution (Figure 3). MBTA faces a different problem: because it does not reallocate tasks, under high loads agents remain committed to tasks that they are no longer suited for, resulting in lengthier delays. Higher loads also mean higher dynamism, resulting in a higher rate of interruption (Figure 4) and corresponding increase in interruption penalties.

One may ask if interruptions are ever desirable, especially since the optimal orderings found by MBTA do not include them. However, this is a consequence of MBTA's inflexible allocation mechanism; a task can only be allocated once. CFLA$^+$ allows dynamic reallocation but its one-step lookahead also precludes interruption. Explicitly disallowing in-

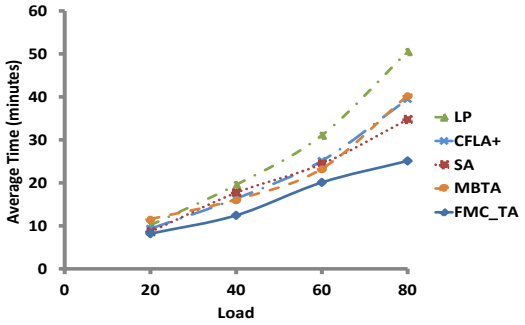

Figure 3: Average execution delay for varying loads.

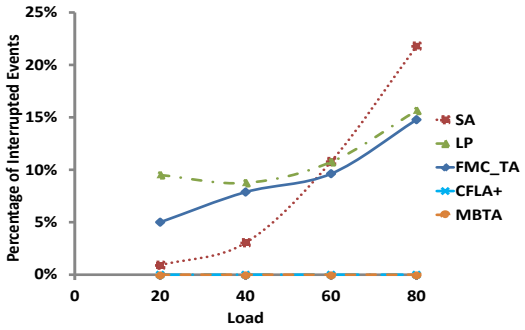

Figure 4: Average percentage of events whose execution was interrupted by at least one agent.

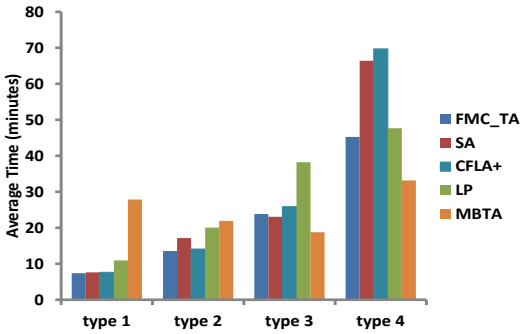

Figure 5: Average arrival time to events of different types.

terruption under FMC_TA resulted in poorer performance.

Rapid response to incidents is highly valued by police departments in its own right, especially for more important events. Agents using FMC_TA result in the lowest overall execution delay over all event types (Figure 3), and are especially responsive to high importance tasks, as shown in Figure 5 for shifts with load of 60. In contrast, SA has much higher delay for low importance tasks, while delays under MBTA are roughly equal for all task types.

FMC_TA achieves low delays by sharing most tasks. Cooperative execution also directly results in higher utility through the capability function, but this is only possible for the 70% of type 1 or 2 tasks; FMC_TA shares a far greater percentage of tasks, as shown in Table 1. Sharing tasks allow agents to divide the workload and thus begin working on subsequent tasks more quickly. SA also benefits from this, but shares more tasks at high loads than FMC_TA. This is detrimental as the greater number of important tasks make

	Load			
Algorithm	20	40	60	80
FMC_TA	100	99.9	95.0	88.9
SA	99.9	99.8	98.3	97.4

Table 1: Percentage of tasks that are shared.

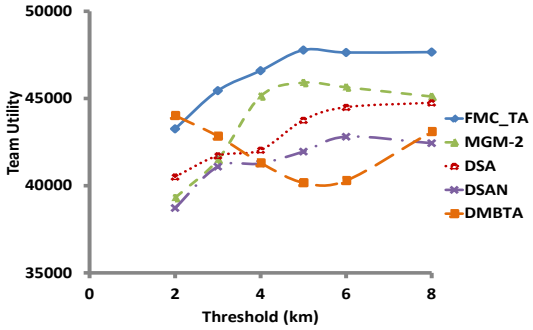

Figure 6: Team utility for varying threshold distances on shifts with 60 events.

the cost of waiting for two agents to move to the location of a shared task comparatively more expensive. The other algorithms do not share tasks.

Evaluation of Distributed Algorithms

We compared the distributed version of FMC_TA to DMBTA (a distributed version of MBTA) as well as general-purpose algorithms solving a distributed constraint optimization problem (DCOP) formulation of LEP. Each agent in all algorithms only knew of tasks in its local environment, i.e., tasks within the threshold distance of the agent.

In the DCOP, each agent a_i holds a sequence of variables $z_{i_1}, ..., z_{i_k}$, where k is the maximum number of fractions of tasks an agent can be allocated. The domain of each variable is the set of tasks within the threshold distance, and a patrol task. The order of the variables represents the order in which the agent will perform these tasks, i.e., a_i will first perform the fraction of task assigned to variable z_{i_1}, then the fraction of task assigned to z_{i_2} etc. Utilities were represented by a constraint between all variables for each task. There are also constraints limiting each agent to one patrol task, which must be scheduled after all events.

Results for three DCOP algorithms are presented: the Distributed Stochastic Algorithm (DSA) (Zhang et al. 2005), Distributed Simulated Annealing (DSAN) (Arshad and Silaghi 2004), and MGM-2 (Maheswaran et al. 2004). These algorithms are known to produce high quality results while (in contrast to algorithms like Max-sum and DALO) their running time does not increase exponentially with the number of agents involved in each constraint. This is extremely important because in LEP, constraints generally involve many agents. The comparison algorithms are local-search algorithms that iteratively improve a selected assignment. The agents running each algorithm select their previous allocation as their initial assignment following a dynamic event. Following each dynamic event, the agents performed 400 iterations of the DCOP algorithm before deciding on the allocation.

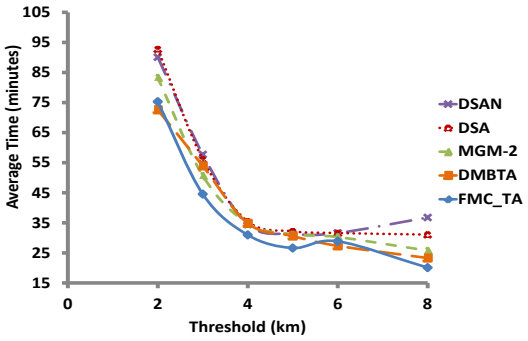

Figure 7: Average execution delay for varying threshold distances, for shifts with 60 events.

Figure 6 presents the team utility for threshold distances from 2 km to 8 km for shifts with load of 60. By comparing with Figure 2, it is clear that the distributed algorithms find high quality solutions, but that FMC_TA dominates, achieving higher utility under almost all thresholds (the difference between FMC_TA and DMBTA for the 2-km-threshold is not statistically significant). The utility under FMC_TA plateaus after a threshold of 5 km, finding solutions of statistically equal quality to that of the centralized algorithm. In contrast, DMBTA, which does relatively well for low thresholds, does poorly at higher thresholds.

FMC_TA also finds solutions with the lowest average task execution delay, as shown in Figure 7 (FMC_TA and DMBTA are statistically equal with the threshold of 6 km). FMC_TA was especially successful in prioritizing important tasks to minimize execution delay, while the delays for tasks of all types were roughly equal in schedules found by the other algorithms (figure omitted for lack of space).

Conclusion and Future Work

Our experimental results show that FMC_TA effectively and efficiently allocates and schedules tasks for agents in the complex, dynamic settings characteristic of law enforcement, which is common in other multi-agent applications. The comparison with LP demonstrates that the two-stage allocate-schedule approach is not solely responsible for the high quality of its solutions. Instead, it is the combination of fairness and efficiency of the allocations in the restricted problem that result in allocations that share important tasks, enabling synergistic cooperation that leads to higher quality task execution while driving down delays.

Despite this success, our results also pointed to a potential shortcoming: utility decreased in settings with high dynamism and many tasks. This effect was not limited to FMC_TA but afflicted many other, unrelated algorithms. This may stem from our assumption that the future is unpredictable; while this is a very general approach, it precludes reasoning about what dynamic changes may occur in the future. Learning a model of the dynamics has been successfully applied in market-based task allocation before (Jones, Dias, and Stentz 2007), and incorporating such reasoning into FMC_TA is an area of future research.

References

Arshad, M., and Silaghi, M. C. 2004. Distributed simulated annealing. *Distributed Constraint Problem Solving and Reasoning in Multi-Agent Systems, Frontiers in Artificial Intelligence and Applications series* 112.

Devanur, N. R.; Papadimitriou, C. H.; Saberi, A.; and Vazirani, V. V. 2002. Market equilibrium via a primal-dual-type algorithm. In *FOCS '02: Proceedings of the 43rd Symposium on Foundations of Computer Science*, 389–395.

Farinelli, A.; Rogers, A.; Petcu, A.; and Jennings, N. R. 2008. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-08)*, 639–646.

Gale, D. 1960. *The Theory of Linear Economic Models*. McGraw-Hill.

Jones, E. G.; Dias, M. B.; and Stentz, A. 2007. Learning-enhanced market-based task allocation for oversubscribed domains. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Macarthur, K.; Stranders, R.; Ramchurn, S.; and Jennings, N. 2011. A distributed anytime algorithm for dynamic task allocation in multi-agent systems. In *Proceedings of the 25th Conference on Artificial Intelligence (AAAI-11)*, 701–706. AAAI Press.

Maheswaran, R. T.; Tambe, M.; Bowring, E.; Pearce, J. P.; and Varakantham, P. 2004. Distributed algorithms for DCOP: A graphical-game-based approach. In *Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems*, 15–17.

Poole, D., and Mackworth, A. K. 2010. *Artificial Intelligence - Foundations of Computational Agents*. Cambridge University Press.

Ramchurn, S. D.; Polukarov, M.; Farinelli, A.; Truong, C.; and Jennings, N. R. 2010a. Coalition formation with spatial and temporal constraints. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-10)*, 1181–1188.

Ramchurn, S. D.; Farinelli, A.; Macarthur, K. S.; and Jennings, N. R. 2010b. Decentralized coordination in RoboCup Rescue. *The Computer Journal* 53(9):1447–1461.

Reeves, C. R., ed. 1993. *Modern heuristic techniques for combinatorial problems*. New York, NY, USA: John Wiley & Sons, Inc.

Reijnierse, J. H., and Potters, J. A. M. 1998. On finding an envy-free Pareto-optimal division. *Mathematical Programming* 83:291–311.

Schoneveld, A.; de Ronde, J. F.; and Sloot, P. M. A. 1997. On the complexity of task allocation. *Journal of Complexity* 3:52–60.

Walsh, W., and Wellman, M. 1998. A market protocol for decentralized task allocation. In *Proceedings of the International Conference on Multi-Agent Systems*, 325 –332.

Zhang, W.; Xing, Z.; Wang, G.; and Wittenburg, L. 2005. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraints optimization problems in sensor networks. *Artificial Intelligence* 161:1-2:55–88.

Zhang, L. 2011. Proportional response dynamics in the Fisher market. *Theoretical Computer Science* 412(24):2691–2698.