Multi-Robot Task Scheduling

Yu ("Tony") Zhang   Lynne E. Parker

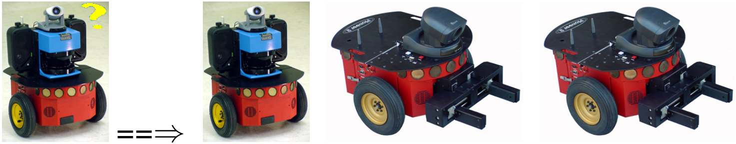
Distributed Intelligence Laboratory
Electrical Engineering and Computer Science Department
University of Tennessee, Knoxville TN, USA

IEEE International Conference on
Robotics and Automation, 2013

Multi-robot task scheduling

Multi-robot tasks:

Individual robots may not have all the required capabilities



Scheduling:

- A set of robots, $R = \{r_1, ..., r_i, ...\}$
- A set of tasks, $T = \{t_1, ..., t_l, ...\}$

Build a schedule to optimize a *func*, $\{R_l, s_l, p_l\}_l$

Multi-robot task scheduling

To represent a general scheduling problem: $P|T|func$

Multi-robot task scheduling:

- $P \rightarrow$ Multi-purpose processor
- $T \rightarrow$ Multi-processor task
- Restrictions:
  - Execution is non-preemptive
  - Robots are non-divisible

or the *MPM MPT* problem [Gerkey and Mataric, 2004]

Complexity of *MPM MPT*

With *func* $= \sum_l e_l$:

- *MPM*: polynomial-time solvable
- *MPT*: $\mathcal{NP}$-hard
- *MPT*2: $\mathcal{NP}$-hard

Two types of multi-robot tasks:

- Loosely coupled: reducible to single robot tasks (*MPM MPT* becomes *MPM*)
- Tightly coupled: ?

Efficient algorithms, preferably with solution bounds, are needed.

Scheduling for tightly coupled multi-robot tasks

Steps:

1. Reduce *MPM MPT* to *MPM*
2. Solve the *MPM* problem

When considering a coalition as a robot, *MPM MPT* becomes *MPM*.

However, coalitions can interfere with each other:

Coalition 1: $\{r_1, r_4, r_5\}$

Coalition 2: $\{r_4, r_6\}$

Contributions

- Considers the scheduling problem for multi-robot tasks at the coalition level
- Proposes four efficient heuristics to address the problem with provable solution bounds
- Provides formal analyses and simulation results to demonstrate and compare their performances

Notations

Table: NOTATIONS USED

R	Set of robots	r_i
C	Set of coalitions	c_j
T	Set of tasks	t_l
p_{jl}	Processing time of t_l by c_j	
e_l	End time of task t_l	

We consider $func = \sum_l e_l$

MinProcTime

Definition (*MinProcTime*)

At each step:

1. Find the assignment that has the smallest p_{jl}
2. Schedule the task at the earliest possible time

Theorem

The MinProcTime heuristic yields a solution quality bounded by $\frac{|T|+1}{2}$.

Tight solution bound

MinStepSum

Definition (*MinStepSum*)

At each step:

1. Find the assignment that increases $\sum_l e_l$ the least

Theorem

The MinStepSum heuristic yields a solution quality bounded by $\frac{|T|+1}{2}$.

Tight solution bound

InterfereAssign

To consider the interference between coalitions:

Definition (*Coalition Interference*)

For any two coalitions c_j and $c_{j'}$ ($j \neq j'$), c_j interferes (or conflicts) with $c_{j'}$ if and only if $c_j \cap c_{j'} \neq \emptyset$.

Consider the impact of an assignment $c_j \rightarrow t_l$ on $\sum_l e_l$:

1. The assignment's processing time p_{jl}
2. Tasks that are scheduled on c_j after t_l
3. Tasks scheduled on coalitions that interfere with c_j

InterfereAssign

For $c_j \rightarrow t_l$:

1. The assignment's processing time p_{jl}
2. Tasks that are scheduled on c_j after t_l

Together, contribute $I_{jl} \cdot p_{jl}$

I_{jl}: scheduling position for t_l on c_j

For example:

$c_2 : t_2 \Rightarrow t_1 \Rightarrow t_3$

For t_2, $I_{22} = 3$ (including influence on t_1 and t_3)

InterfereAssign

For $c_j \rightarrow t_l$:

3. Tasks scheduled on coalitions that influence with c_j

Upper bound is $|\cup_{c \in F_j} N_c| \cdot p_{jl}$

F_j: coalitions that interfere with c_j

N_c: set of tasks that c can accomplish

InterfereAssign

Convert *MPM MPT* to *MPM* by constructing an assignment problem:

- Create a task node for each task t_l
- Create a coalition-position node for each coalition c_j and position pair, with positions ranging from 1 to N_{c_j} for coalition c_j
- If a coalition c_j can accomplish a task t_l, connect t_l with all coalition-position nodes for c_j, and set the weights to be $(|\cup_{c \in F_j} N_c| + I_{jl}) \cdot p_{jl}$, respectively, based on I_{jl}

Now, solve this problem optimally.

InterfereAssign

Lemma

There exists a schedule that is no worse than the solution of the assignment problem.

Theorem

The schedule that is constructed from the solution of the assignment problem yields a solution quality bounded by $\max_j |\cup_{c \in F_j} N_c| + 1$.

- Quality dependent on complex structure of the problem instance
- Less coalition interference, better quality
- Optimal solution for single robot tasks

MinInterfere

In *InterfereAssign*:

- $|\cup_{c \in F_j} N_c|$ is an overestimation

Definition (*MinInterfere*)

At each step:

1. Compute β_{jl} and choose the assignment that minimizes it:

$$\beta_{jl} = e_{jl} + |\cup_{c \in F_j} N_c \setminus M_{jl}| \cdot p_{jl}$$

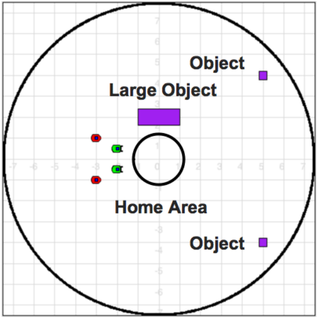
M_{jl}: $t_l \cup$ the set of tasks that are scheduled before $c_j \rightarrow t_l$

Summary

Table: SUMMARY OF DISCUSSED HEURISTICS

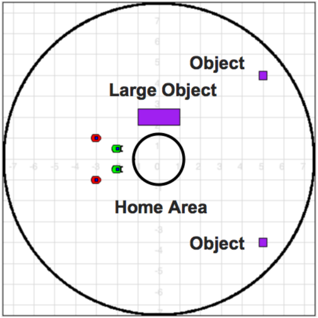
Name	**Solution Bound**	**Complexity**
Optimal	1	$O((\|C\|\|T\|)^{\|T\|}\|T\|!)$
MinProcTime	$\frac{\|T\|+1}{2}$	$O(\|C\|\|T\|^3)$
MinStepTime	$\frac{\|T\|+1}{2}$	$O(\|C\|\|T\|^3)$
InterfereAssign	$\max_j \|\cup_{c \in F_j} N_c\| + 1$	$O(\|C\|^3\|T\|^3)$
MinInterfere	Not Determined	$O(\|C\|\|T\|^3)$

A simple scenario



Task	Robots Required	Process Time
1) Object 1	One gripper, one localizer	6
2) Object 2	One gripper, one localizer	6
3) Large Object	Two grippers	5

A simple scenario



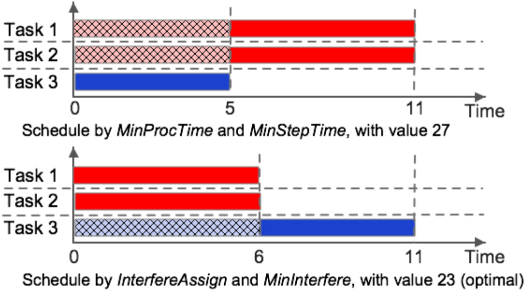


Figure: Schedules created by our heuristics

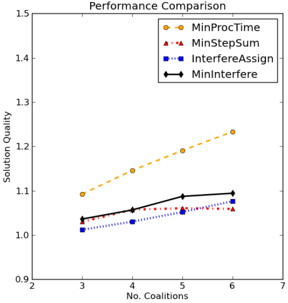
Heuristics that consider the interference produce the optimal solution

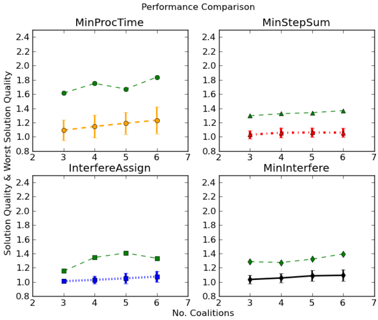
Parameters

Table: PARAMETERS USED IN THE SIMULATIONS

Parameter	Description
n_c	No. of coalitions
n_t	No. of tasks
n_f	Average no. of conflicting coalitions per coalition
n_e	Average no. of executable tasks per coalition
n_{min}, n_{max}	Minimum and maximum processing time

Varying n_c

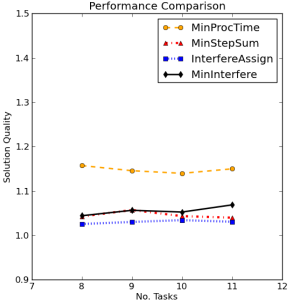


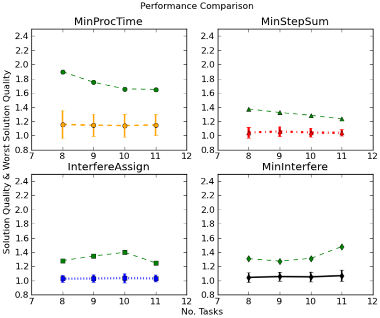


The average solution quality is better than the proven bounds

MinStepSum, *InterfereAssign* and *MinInterfere* perform similarly; *InterfereAssign* is better for smaller n_c (i.e., $3-4$)

Varying n_t

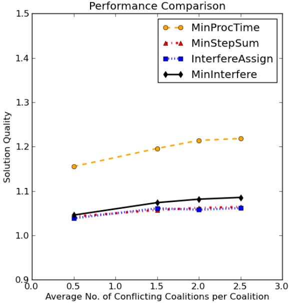


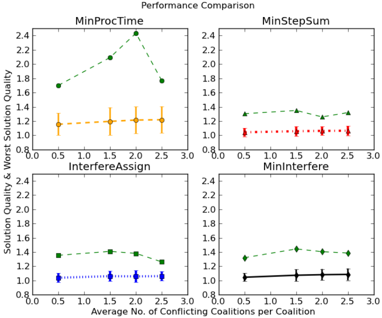


Similar observations

Since n_f stays as a constant, the curve formed is smoother

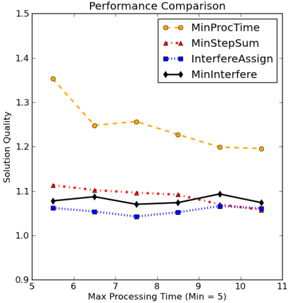
Varying n_f

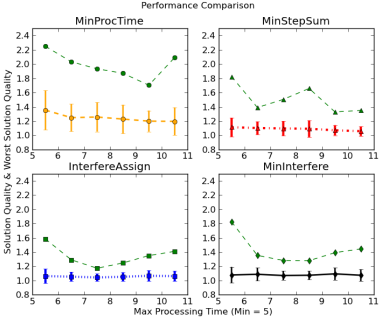




Performance decreases as the interference becomes more complex

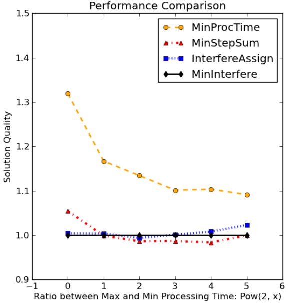
Varying n_{max}

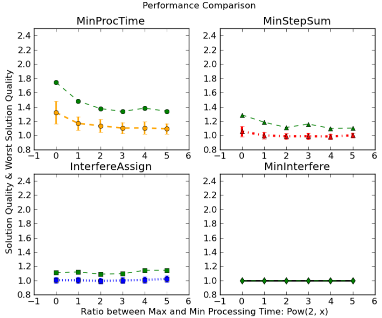




Increase of n_{max} does not always decrease the performance

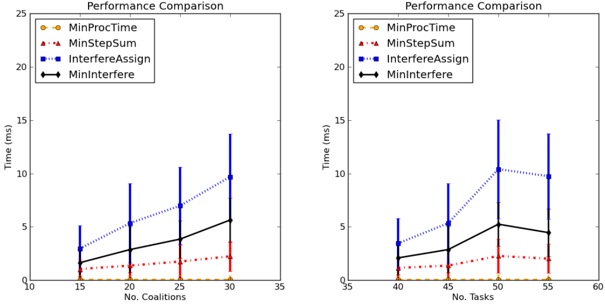
Varying n_{max}, with large n_c and n_t





MinStepSum performs slightly better with large n_c and n_t

Time analysis. Left: Varying n_c. Right: Varying n_t



Has potentials to be applied to large-size problems

Conclusions

- When there is less interference between coalitions, use InterfereAssign
- Otherwise, choose the best

Contributions

- Considers the scheduling problem for multi-robot tasks at the coalition level
- Proposes four efficient heuristics to address the problem with provable solution bounds
- Provides formal analyses and simulation results to demonstrate and compare their performances

References

Gerkey, B. and Mataric, M. (2004).
A formal analysis and taxonomy of task allocation in multi-robot systems.
The International Journal of Robotics Research, 23(9):939–954.

Zhang, Y. and Parker, L. (2010).
IQ-ASyMTRe: Synthesizing coalition formation and execution for tightly-coupled multirobot tasks.
In *IEEE/RSJ Int'l Conference on Intelligent Robots and Systems*.