Multi-Robot Task Allocation for logistic applications

Davide Zorzi - VR414572

March 21, 2019

University of Verona

Industrial Logistics

The **industrial logistics** is the process of **planning**, **organization** and **control** of all the activities of handling and **storage** of goods, guarantee an adequate level of **service** to the customer consistent with the **costs** to it associated.

Multi-Robot Systems for logistic applications

Kiva warehouse-management system.

Thesis contribution

The contribution of this thesis:

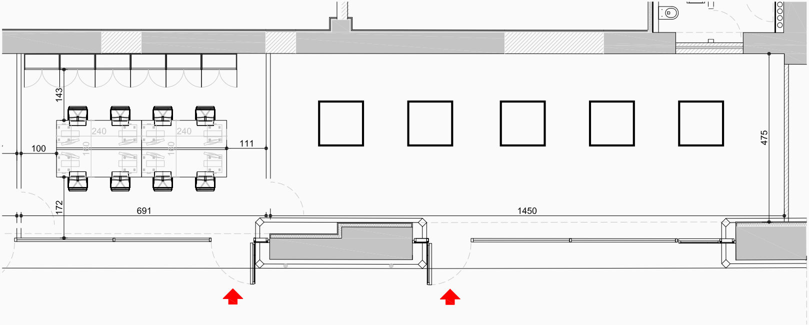
\+ increasing **productivity**

− **time** and **distance** travel

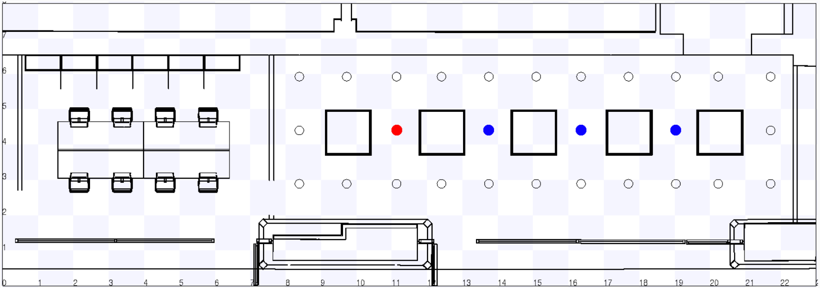
How to **partition** a finite set of task $\mathcal{T}$

- **Baseline** for our experiments
  1. Single robot : Single task (SR:ST)
- Proposing two methods
  1. Set Partition Strategy - Single robot : Multiple task (SPS1:N)
  2. Greedy Set Partition Strategy - Single robot : Multiple task (GSP1:N)
- **real scenario**: Computer Engineering for Industry 4.0 Laboratory (ICE Lab)
- extension of `ROS` package

ICE Laboratory



ICE Laboratory for logistic application

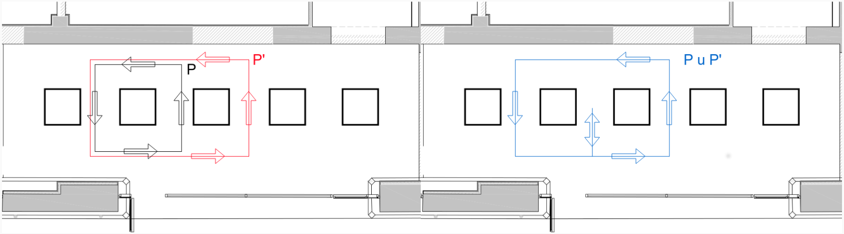


- Loading bay
- Unloading bays
- Vertices

Problem formalization

Given a set of tasks $\mathcal{T}$ it defines intrinsically a set of orders O. One order perform a subset S of $\mathcal{T}$, $S \subseteq \mathcal{T}$.

$S = \{T_1, \cdots, T_k\}$ for each element we **combine** their paths P to form a single path $\pi = \{v_1, \cdots, v_i\}$.



Problem formalization 2

We **maximize** the total demand (d_S).

$$d_S = demand(T_1) + \cdots + demand(T_k)$$

The heuristic function $v(\cdot)$ which can be defined for any task T or subset S:

$$v(S) = \frac{f(\pi)}{d_S}$$

For compute the **best partition** the heuristic is based on the concept of **loss** L, which can be defined for any pair of subset S_i, S_j as:

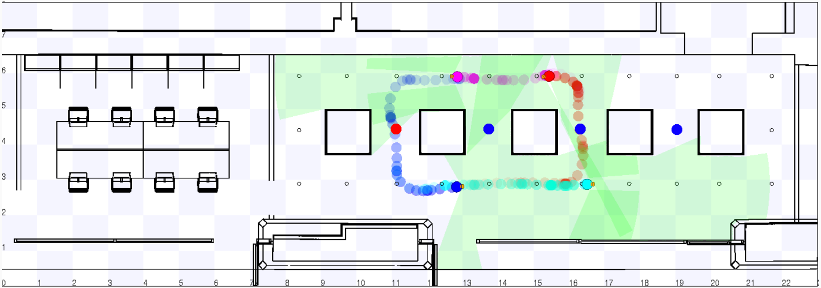
$$L(S_i, S_j) = v(\{S_i \cup S_j\}) - v(S_i) - v(S_j)$$

We want **minimize** the cost:

$$L(S_i, S_j) < 0$$

Single robot : Single task (SR:ST)

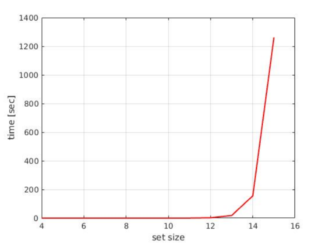
This method is a **baseline** for our logistic scenario.



The important constraint of this approach is to consider only **one task** allocated for **one robot** at time.

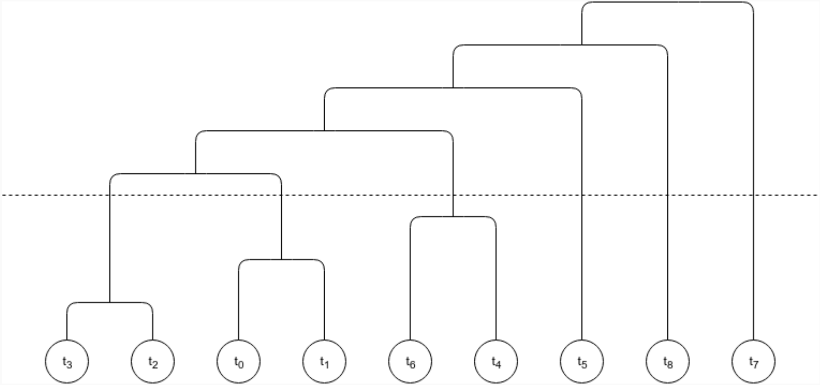
Set Partition Strategy - Single robot : Multiple task (SPS1:N)

iteration	partition size	partition
1	1	{{a, b, c, d}}
2	2	{{a, b, c}, {d}}
3	2	{{a, b, d}, {c}}
4	2	{{a, b}, {c, d}}
5	3	{{a, b}, {c}, {d}}
6	2	{{a, c, d}, {b}}
7	2	{{a, c}, {b, d}}
8	3	{{a, c}, {b},{d}}
9	2	{{a, d}, {b, c}}
10	2	{{a}, {b, c, d}}
11	3	{{a}, {b, c}, {d}}
12	3	{{a, d}, {b}, {c}}
13	3	{{a}, {b, d}, {c}}
14	3	{{a}, {b}, {c, d}}
15	4	{{a}, {b}, {c},{d}}



Greedy Set Partition Strategy - Single robot : Multiple task (GSP1:N)

The main concept of this approach is composing tasks using Greedy **Coalition Formation** strategy.



The horizontal line represents a cut during execution it defines the coalition structure.

Example

Given a set of tasks $\mathcal{T} = \{\{T_0\}, \{T_1\}, \cdots, \{T_8\}\}$ defined like:

$$T_i = (item, demand, unloading\_bay).$$

The agents have the **same capacity** $C_{0,1,2,3} = 4$.

task	item	demand	unloading bay
0	A	1	0
1	B	2	1
2	C	3	2
3	A	1	0
4	B	2	1
5	C	3	2
6	A	1	0
7	B	2	1
8	C	3	2

Often in the logistic environments robots are **all equal**.

Example 2

Result SPS:

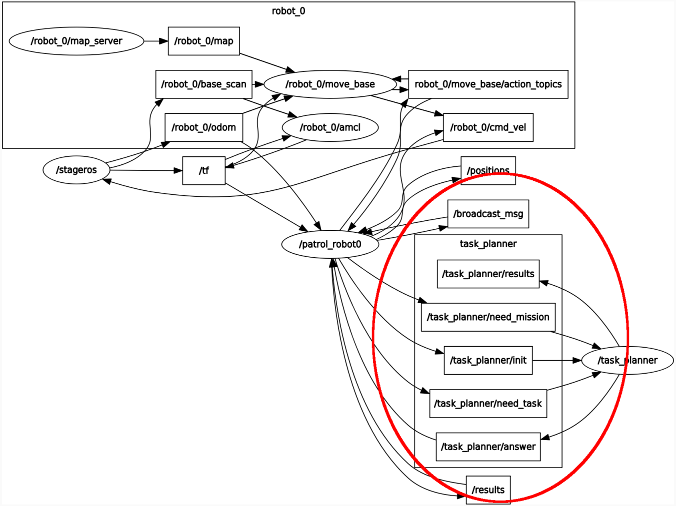
task	item	demand	unloading bay
{4,7}	B	4	1
{0,1,3}	{A,B}	4	{0,1}
{2,6}	{C,A}	4	{0,2}
5	C	3	2
8	C	3	2

Result GSP:

task	item	demand	unloading bay
{3,2}	{A,C}	4	{0,2}
{0,1}	{A,B}	3	{0,1}
{6,4}	{A,B}	3	{0,1}
5	C	3	2
8	C	3	2
7	B	2	1

Video

ROS package Logistic_sim



Results

Configuration	Algorithm	$\overline{Time}$	$\overline{Interference}$	$\overline{Distance}$	$\bar{\sigma}(Distance)$
6/-/2	**SR:ST**	218.32[±6.19]	63.45	3747.90	87.8
6/3/2	GSP1:N	194.52[±6.42]	49.65	3401.15	251.37
	SPS1:N	177.00[±1.99]	49.34	3132.5	**0**
6/5/2	GSP1:N	142.08[±1.39]	42.2	2714.25	206.43
	SPS1:N	138.98[±2.41]	39.38	2601.25	156.47
6/-/4	**SR:ST**	124.52[±3.12]	42	2194.75	114.2
6/3/4	GSP1:N	117.44[±1.85]	35.75	1769	43.83
	SPS1:N	115.28[±4.10]	33.5	1702.5	23.67
6/5/4	GSP1:N	93.4[±1.01]	29	1688.5	34.5
	SPS1:N	91.8[±2.14]	30.75	1546.5	35.8
9/-/2	**SR:ST**	292.24[±3.06]	85.5	5201.5	34.76
9/3/2	GSP1:N	265.72[±2.64]	71.5	4491.5	**0**
	SPS1:N	240.74[±10.42]	75.5	4232.5	310.43
9/5/2	GSP1:N	232.84[±4.71]	68.85	4041.25	236
	SPS1:N	168.34[±2.03]	50.5	3132.5	**0**
9/-/4	**SR:ST**	178.55[±4.23]	52	2755.75	135.8
9/3/4	GSP1:N	152.55[±2.87]	46.75	2200	113.4
	SPS1:N	134.23[±3.25]	40.63	2182.5	27
9/5/4	GSP1:N	134.23[±3.26]	40.6	2098.3	93.45
	SPS1:N	93.05[±5.15]	32.25	1530.25	**0**
21/-/2	**SR:ST**	629.10[±8.84]	154.6	11773.5	229.75
21/3/2	GSP1:N	561.93[±8.00]	134.3	10133.16	201.2
21/5/2	GSP1:N	497.45[±6.15]	126	9079	210.4
21/-/4	**SR:ST**	402.12[±5.06]	132.25	6232.35	295.1
21/3/4	GSP1:N	343.23[±6.10]	98.23	5231.25	342.2
21/5/4	GSP1:N	294.40[±7.60]	77.63	4683.25	367.5

Conclusions and Future Work

In conclusion:

- The results respects the initial expectations.
- The quality of solutions found by GSP is comparable with the quality of solutions found by SPS.
- Coalition Formation problem can approximate the results of a set partion problem in **less** time complexity.

I am focused on a **centralized coordinator** in the future works I want to perform a **distributed coordiantion**.

That strategy should be more **flessible**, **adaptive** at the situation on the traveling orders then **fault-tolenace**.

Thank you for your attention!