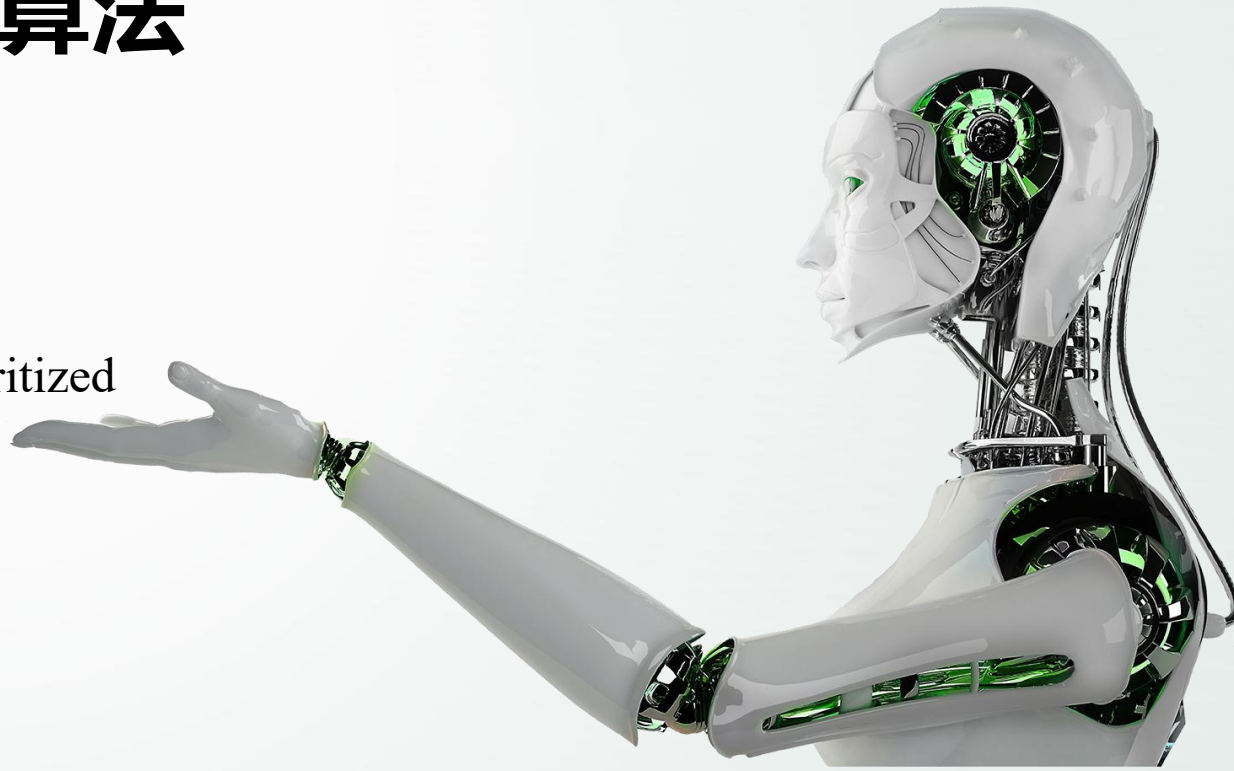
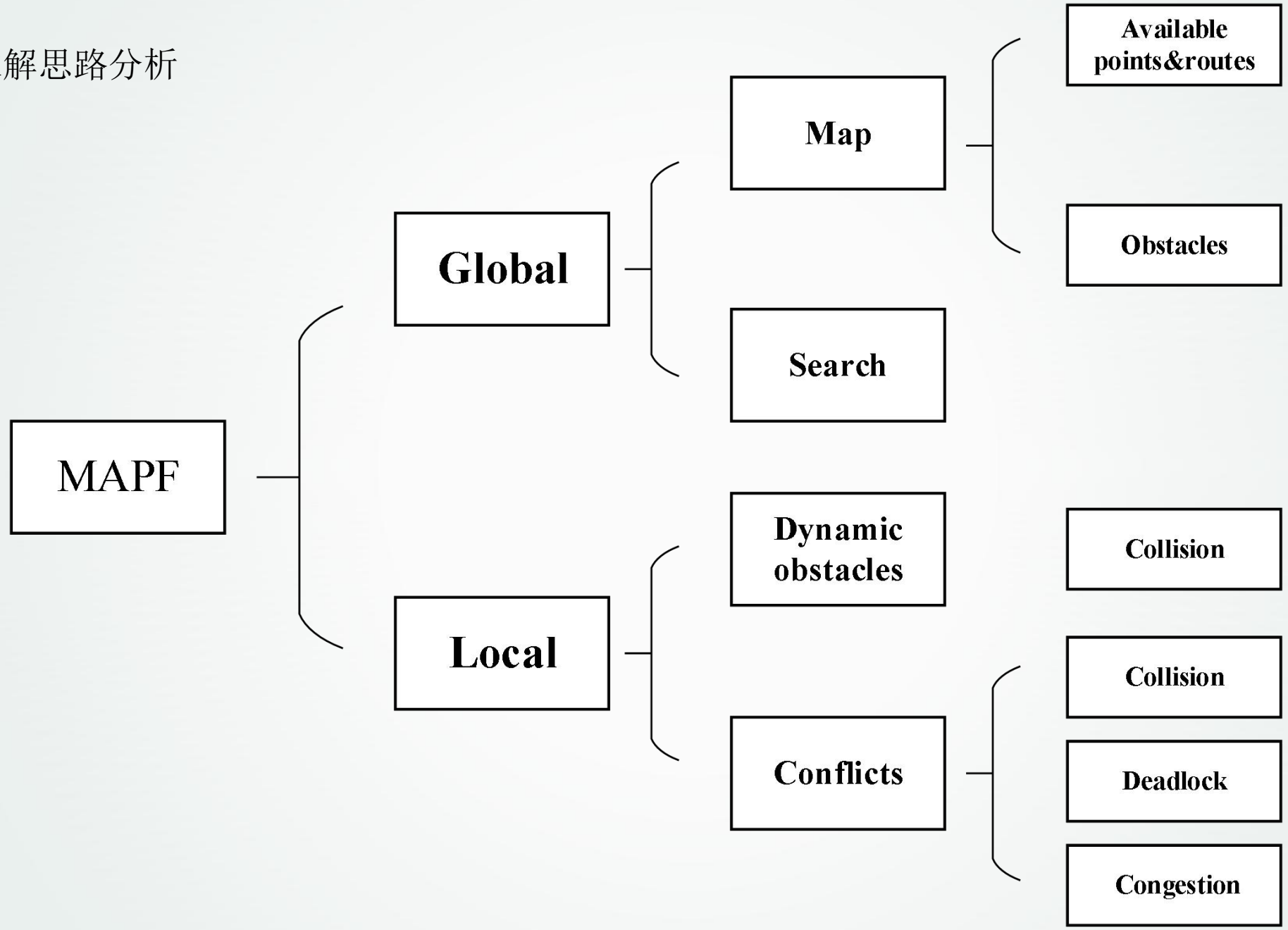
第六章

MAPF主动避障-PP算法

1. Prioritized Planning
2. Revised Prioritized Planning
3. Asynchronous Decentralized Prioritized Planning
4. 实战中的Prioritized Planning



MAPF求解思路分析



1. Prioritized Planning

什么是PP？

On Multiple Moving Objects

Michael Erdmann and Tomás Lozano-Pérez

Artificial Intelligence Laboratory
Massachusetts Institute of Technology

1. Prioritized Planning

In classical prioritized planning each robot is assigned a unique priority. The trajectories for individual robots are then planned sequentially from the highest-priority robot to the lowest priority one. For each robot a trajectory is planned that avoids both the static obstacles in the environment and the higher priority robots moving along the trajectories planned in the previous iterations

a) 每个机器人设置一个优先级
b) 按照优先级由高到低，依次进行规划
c) 优先级低的需要避开优先级高的

解释：
a) 本质上就是给机器人加权重。在Scheduling问题中是比较常用的一种方式，属于排序论范畴，可以看作启发式规则。
b) 这一步把一个多机器人系统的同时规划问题降维至单机器人规划问题
c) 是规则约束的必然结果。

1. Prioritized Planning

PP算法存在的问题

3.3. Prioritized Planning

An alternative approach is to plan motions for several moving objects by planning motions one object at a time. Thus the centralized planning problem is transformed into a series of autonomous planning problems. The appeal of this decomposition approach is that it reduces the problem from a single planning problem in a very high dimensional space to a sequence of planning problems in low dimensional spaces. The disadvantage of this approach is the loss of completeness. By not considering all moving objects at once, the planner runs the risk of choosing a trajectory for an object early on that prevents finding a solution for an object later in the planning sequence.

Any task in which a prioritization of motions may be assigned, may be approached using the decomposition scheme. Examples include tasks in which robots are cooperating in master/slave relationships and tasks in which the order of part assembly is highly constrained. Notice that a prioritization does not imply that an object of lower priority must follow or assist an object of higher priority. In general, an object of low priority may be performing independent operations. A prioritization simply states that the burden of avoiding collisions between two objects falls on the object of lower priority.

1. 把高维空间降为低维空间，减小复杂度
2. 缺少完备性
3. 优先级约束并不意味着低优先级必须服从高优先级
4. 优先级约束只表示避障要求/负担

1. Prioritized Planning

PP算法存在的问题

CENTRALIZED AND DECENTRALIZED ALGORITHMS FOR MULTI-ROBOT TRAJECTORY COORDINATION

Doctoral Thesis

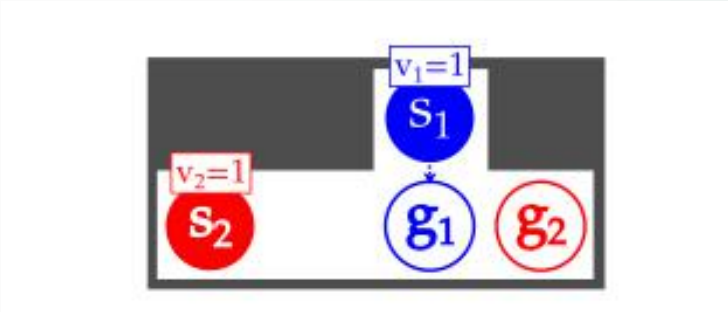
Michal Čáp

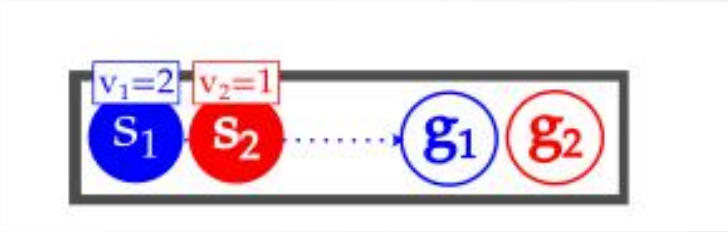
Prague, December 2016

Ph.D. Programme: Electrical Engineering and Information Technology
Branch of study: Artificial Intelligence and Biocybernetics

Supervisor: Prof. Dr. Michal Pěchouček, MSc.

Co-supervisor: Dr. rer. nat. Peter Novák





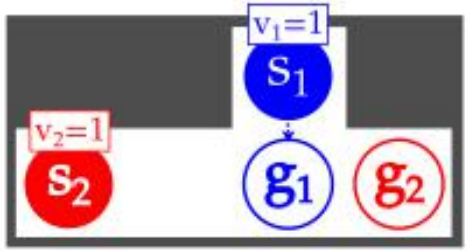
1. Prioritized Planning

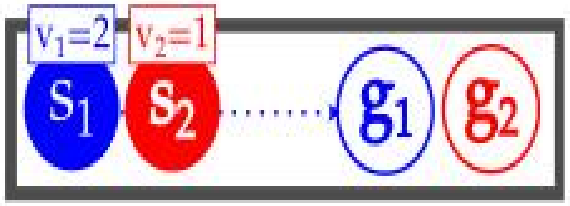
PP算法存在的问题

Let us now analyze when the classical prioritized planning algorithm is bound to fail. The algorithm fails to find a trajectory for robot i if 1) no satisfying path exists for robot i, i.e., the robot cannot reach its destination even if there are no other robots in the workspace; 2) every satisfying trajectory of the robot i is in conflict with some higher-priority robot. There are two types of conflicts that can occur between a satisfying trajectory π of the robot i and a higher-priority robot:

Type A: Occurs if the trajectory π is in conflict with a higher-priority robot which has reached and is "sitting" at its destination, i.e., it is blocked by a static higher-priority robot. Figure 4.1.1 depicts a scenario in which all satisfying trajectories of one of the robots are in Type A conflict.

Type B: Occurs if the trajectory π of robot i is in conflict with a higher-priority robot which is moving towards its destination, i.e., it is "run over" by a moving higher-priority robot. Figure 4.1.2 shows a scenario in which all satisfying trajectories of one of the robots are in Type B conflict.





1)机器人不存在可行的路径，即即使工作空间中没有其他机器人，机器人也无法到达其目的地，则该算法无法为机器人找到轨迹

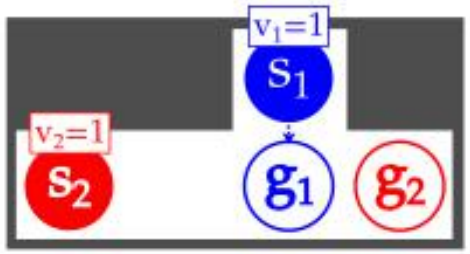
2)机器人的每个可行的轨迹都与某个优先级更高的机器人冲突。

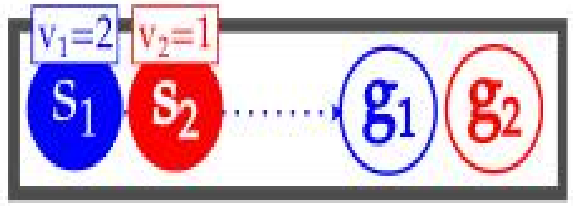
1. Prioritized Planning

PP算法存在的问题

Type A: Occurs if the trajectory π is in conflict with a higher-priority robot which has reached and is "sitting" at its destination, i.e., it is blocked by a static higher-priority robot. Figure 4.1.1 depicts a scenario in which all satisfying trajectories of one of the robots are in Type A conflict.

Type B: Occurs if the trajectory π of robot i is in conflict with a higher-priority robot which is moving towards its destination, i.e., it is "run over" by a moving higher-priority robot. Figure 4.1.2 shows a scenario in which all satisfying trajectories of one of the robots are in Type B conflict.





类型A:如果轨迹π与已经到达并"坐"在目的地的更高优先级机器人冲突，即被静态的更高优先级机器人阻挡

类型B:如果机器人的轨迹π与正在向其目的地移动的较高优先级机器人冲突，即被移动的较高优先级机器人"碾过"，

2. Revised Prioritized Planning

算法设计及改进的一般流程

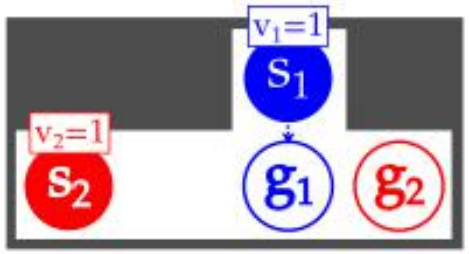
1. 选定算法框架/底层算法
2. 针对所研究的问题对算法进行适当改动
3. 测试并发现问题
4. 针对问题，回溯算法找到原因
5. 根据原因优化和修改算法
6. 测试算法性能，若无问题进行进一步调试得出结果及结论；若有问题回到4

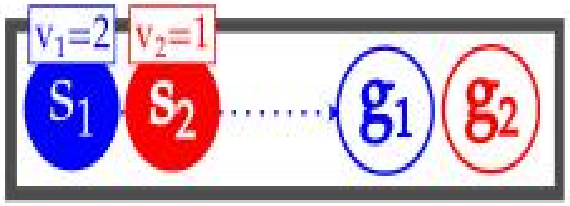
举例

1. 选用GA做为主算法
2. 根据问题的特性，离散或连续，多目标或单目标等特点，修改GA；
3. 测试发现问题（局部最优，收敛快，迭代慢等）
4. 设计local search，调整交叉变异概率，设计启发式编码方式，设计动态fitness，块结构，关键路径等等改进方式；
5. 测试和调试

再比如我们前面讲到的RRT算法及其改进算法RRT*，RRT-Connection等等，就是根据算法的原理决定的算法特性来进行修改。但这并不代表基本算法无用，因为所有的改进算法都是基于基本算法的。它的作用是提供一种思路，一种方式和方法。

2. Revised Prioritized Planning



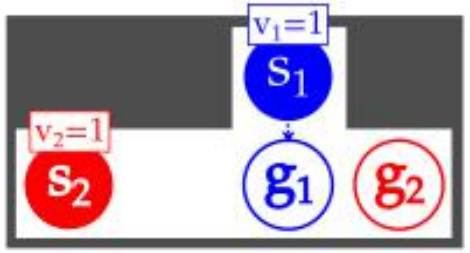


A question that naturally arises is whether it is possible to restrict the class of solvable instances or to alter the classical prioritized planning algorithm in such a way that there is always at least one trajectory without neither Type A nor Type B conflict for each robot.

找到算法存在的问题---是否能够通过改进解决问题？

2. Revised Prioritized Planning

One way to ensure that there is a satisfying trajectory without Type A conflict for every robot is to only consider instances, where each robot has a path to its goal that avoids the goal regions of all higher-priority robots. When each robot follows such a path, then they cannot be engaged in a Type A conflict, because the Type A conflict can only occur at the goal region of one of the higher-priority robots.



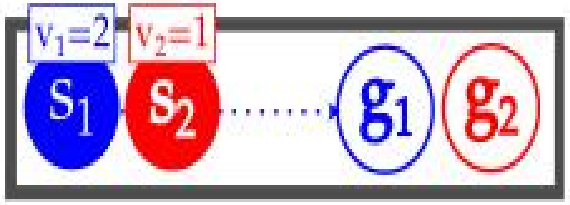
解决type A所示的冲突，确保每个机器人都有一个可行的轨迹：

避开所有高优先级机器人的目标点所在区域。

当每个机器人都遵循这样的路径时，那么它们不再存在类型A冲突。
因为类型A冲突只能发生在其中一个较高优先级机器人的目标区域。

2. Revised Prioritized Planning

One way to ensure that there is a satisfying trajectory without Type B conflict for every robot is to consider only instances in which each robot has a path to its goal that avoids start region of lower-priority robots and enforce that the trajectory of each robot will avoid the start regions of all lower-priority robots. When this is ensured, then any robot always has a fallback option to wait at its start position (since no higher-priority robot can run over its start region) until its desired path is clear of the all higher-priority robots. Thus it can always avoid Type B conflicts.



确保每个机器人都有一个可行的没有B型冲突的轨迹的一种方法

避开低优先级机器人的出发点区域。

任何机器人总是有一个后备选项（缓冲区）在其开始位置等待(因为没有更高优先级的机器人可以在其开始区域上运行），这样可以避免B型冲突。

2. Revised Prioritized Planning

Moreover, if the robot continues by following a path that avoids the goal regions of higher-priority robots, then the resulting trajectory is also guaranteed to avoid Type A conflicts.

The result of the above analysis can be intuitively summarized as follows: If we have a trajectory coordination problem instance in which every robot can reach its goal without crossing a) the regions occupied by the lower-priority robots at their start positions and b) the regions occupied by the higher-priority robots at their goal positions, then such an instance can be in the worst-case resolved by moving the robots sequentially, one after another, to their destinations.

如果机器人继续沿着避开优先级较高的机器人的目标区域的路径前进，那么所得到的轨迹也可以保证避免A型冲突。

因此，针对两类问题，采用的改进方法为：

1. 所有机器人均要避开比其 优先级低的其他机器人的起始点区域
2. 所有机器人均要避开比其优先级高的机器人的目标点区域

2. Revised Prioritized Planning

RPP仍然存在一些问题

无法保证所有机器人都能遵循两条改进规则。原因是实际情况下无法完全实现所有路径无冲突。

再深入分析，本质上还是因为PP算法是一种针对集中式控制系统的规划算法。

集中式有什么缺陷？

依赖控制中心，所有指令都需要通过控制中心下发

3. Asynchronous Decentralized Prioritized Planning

3.1 Synchronous Decentralized Prioritized Planning

集中式方法的缺点之一是，中央控制器必须收集关于系统中每个机器人的当前状态、约束和目标的完整信息，然后才能计算协调解决方案。通过分散算法，这样的信息可以仅通过交换有限数量的共享信息来找到协调解决方案。

如果采用分布式控制，首先可以实现机器人的同步控制和规划。

3. Asynchronous Decentralized Prioritized Planning

3.1 Synchronous Decentralized Prioritized Planning

The algorithm proceeds in synchronized rounds. In every round, each robot ensures that its current trajectory is consistent with the trajectories of higher-priority robots from the previous round. If the current trajectory is consistent, then the robot keeps its current trajectory and remains silent. Otherwise, it finds a new consistent trajectory for itself and broadcasts the trajectory to all other robots. When a robot finishes its computation in the current round, it waits for all other robots to finish the round and all robots simultaneously proceed to the next round. The algorithm successfully terminates if none of the robots changes its current trajectory during a single round. The SD-PP algorithm terminates with failure if there is a robot that fails to find a trajectory that avoids the higher-priority robots following their respective trajectories. The SD-RPP algorithm, on the other hand, finishes with failure if there is a robot that fails to find a satisfying trajectory that avoids the start positions of lower-priority robots. The pseudocode of SD-(R)PP is listed in Algorithm 7.

算法失败的判定：

1. 低优先级机器人无法找到一条能够避开高优先级机器人目标点的可行路径
2. 高优先级机器人无法找到一条能够避开低优先级机器人起始点的可行路径

1. 所有机器人同时运动
2. 实时共享各自的路径信息
3. 如果高优先级机器人暂时未改变路径，比其低优先级的其他机器人也不需改变路径
4. 如果高改变，则低全部改变
5. 当所有机器人的路径均不再变化时，算法终止

3. Asynchronous Decentralized Prioritized Planning

3.1 Synchronous Decentralized Prioritized Planning

SD-RPP是在RPP基础上，把集中式改为分布式，达到两个效果：

1. 同步规划
2. 效率提升

效率能否进一步提升？

Algorithm 7: Synchronized Decentralized Implementation of (Revised) Prioritized Planning. Pseudocode for robot i

Algorithm SD-(R)PP
  $\pi_i \leftarrow \emptyset$;
  $H_i \leftarrow \emptyset$;
  $S \leftarrow \begin{cases} \emptyset & \text{for SD-PP} \\ \bigcup_{j>i} S^j & \text{for SD-RPP} \end{cases}$;
  repeat
    $\pi^* \leftarrow$ Find-consistent$(\pi_i, \mathcal{W} \setminus S, \Delta(H_i))$;
    if $\pi^* = \emptyset$ **then**
      report failure and terminate;
    else if $\pi^* \neq \pi_i$ **then**
      $\pi_i \leftarrow \pi^*$;
      broadcast INFORM$(i, R_i^{\Delta}(\pi^*))$;
    wait for INFORM messages from all other robots, wait for all other robots to finish processing INFORM messages ;
  until *not global termination detected*;
Handle-message INFORM(j, Δ_j)
  if $j < i$ **then**
    $H_i \leftarrow \left(H_i \setminus \{(j, \Delta_j') : (j, \Delta_j') \in H_i\}\right) \cup \{(j, \Delta_j)\}$;
Function Find-consistent$(\pi, \mathcal{W}, \Delta)$
  if $\pi = \emptyset \vee \neg$consistent$_i(\pi, \Delta)$ **then**
    $\pi^* \leftarrow$Best-traj$_i(\mathcal{W}, \Delta)$;
    return π^*;
  else
    return π;

3.2 Asynchronous Decentralized Prioritized Planning

Due to its synchronous nature, the SD-(R)PP algorithm does not fully exploit the computational resources distributed among individual robots. In every iteration, the robots that finished their trajectory planning routine earlier, or did not have to re-plan at all, stay idle while waiting for the slower computing robots in that round. However, they could use the time to resolve some of the conflicts among themselves and speed up the overall process. An example of a situation, where the asynchronous algorithm would be beneficial is illustrated in Figure 5.2.1.

由于同步性，SD-(R)PP算法没有充分利用分布在各个机器人之间的计算资源。在每次迭代中，提前完成轨迹规划例程的机器人，或者根本不需要重新规划的机器人，都会在等待下一轮计算速度较慢的机器人时保持空闲。然而，他们可以利用这段时间解决他们之间的一些冲突，加快整个进程。

3. Asynchronous Decentralized Prioritized Planning

3.2 Asynchronous Decentralized Prioritized Planning

To deal with such an inefficiency, we propose an asynchronous decentralized implementation of classical prioritized planning, abbreviated AD-PP, and revised prioritized planning scheme, abbreviated AD-RPP. The AD-(R)PP abbreviation is used when a statement holds for both variants.

The pseudocode of AD-(R)PP is shown in Algorithm 8. The asynchronous algorithm replaces the concept of globally synchronized rounds (while loop in Algorithm 7) by a reactive approach in which every robot reacts merely to incoming INFORM messages. Upon receiving an INFORM message (**Handle-message** INFORM(j, Δ_j) routine in Algorithm 8), the robot simply replaces the information about the trajectory of the sending robot in its trajectory store and checks whether its current trajectory is still consistent with the new contents of its trajectory store. If the current trajectory is inconsistent, the robot triggers replanning and informs other robots about its new trajectory. Otherwise, the robot keeps its current trajectory and remains silent.

1. 异步算法
2. 反应式方法取代了全局同步循环
3. 每个机器人只对传入的通知消息做出反应
4. 如果当前轨迹不一致，机器人会触发重新计划，并通知其他机器人其新轨迹。
5. 如果轨迹不变，机器人保持其当前轨迹并保持沉默。

AD-RPP和SD-RPP最大的不同是，AD-RPP不需要实时同步所有机器人的状态再决定路径的变更与否，只在某一个或某几个机器人需要变更路径时发送信息。这样可以避免无效搜索。

3. Asynchronous Decentralized Prioritized Planning

3.2 Asynchronous Decentralized Prioritized Planning

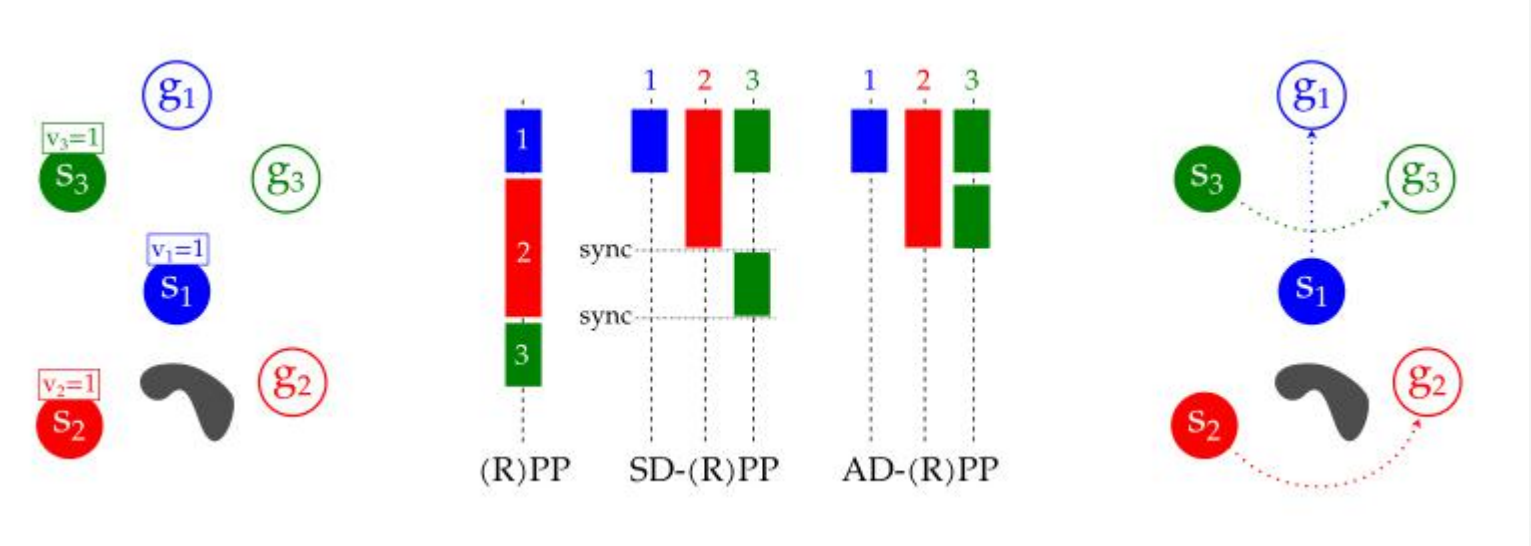
Algorithm 8: Asynchronous Decentralized Implementation of (Revised) Prioritized Planning

Algorithm AD-(R)PP
- $\pi_i \leftarrow \emptyset$;
- $H_i \leftarrow \emptyset$;
- $S \leftarrow \begin{cases} \emptyset & \text{for AD-PP} \\ \bigcup_{j>i} S^j & \text{for AD-RPP} \end{cases}$;
- $\pi_i \leftarrow$ `Find-consistent` $(\pi_i, \mathcal{W} \backslash S, \Delta(H_i))$;
- **if** $\pi_i = \emptyset$ **then**
  - report failure and terminate;
- **else**
  - **broadcast** INFORM$(i, R_i^{\Delta}(\pi_i))$;
  - **wait** for global termination;

Handle-message INFORM(j, Δ_j)
- **if** $j < i$ **then**
  - $H_i \leftarrow \left(H_i \setminus \{(j, \Delta_j') : (j, \Delta_j') \in H_i\}\right) \cup \{(j, \Delta_j)\}$;
  - $\pi^* \leftarrow$ `Find-consistent` $(\pi_i, \mathcal{W} \setminus S, \Delta(H_i))$;
  - **if** $\pi^* = \emptyset$ **then**
    - report failure and terminate;
  - **else if** $\pi^* \neq \pi_i$ **then**
    - $\pi_i \leftarrow \pi^*$;
    - **broadcast** INFORM$(i, R_i^{\Delta}(\pi^*))$;

3. Asynchronous Decentralized Prioritized Planning

3.2 Asynchronous Decentralized Prioritized Planning



异步提升效率，反映在算法表现上，就是加速收敛

4. 实战中的Prioritized Planning

Prioritized Motion Planning for Multiple Robots

Jur P. van den Berg Mark H. Overmars
Institute of Information and Computing Sciences
Utrecht University, The Netherlands
{*berg, markov*}*@cs.uu.nl*

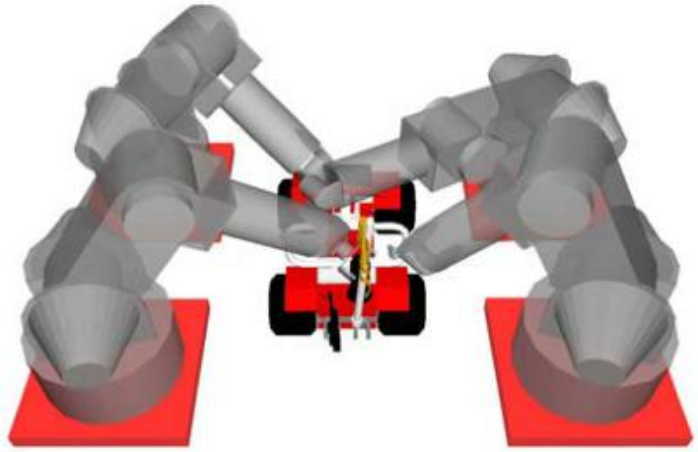


Fig. 1. An environment with four articulated robots manipulating a car.

4. 实战中的Prioritized Planning

2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Asynchronous Decentralized Prioritized Planning for Coordination in Multi-Robot System

Michal Čáp, Peter Novák, Martin Selecký, Jan Faigl, Jiří Vokřínek

Fig. 5: The UAV during a field experiment.

作业

下载并调试以下链接的代码（博士论文，文档P79，论文P67）

https://github.com/mcapino/adpp-journal

要求：

1. 跑通即可
2. 写出该算法的伪代码