

# Task Monitoring and Rescheduling for Opportunity and Failure Management

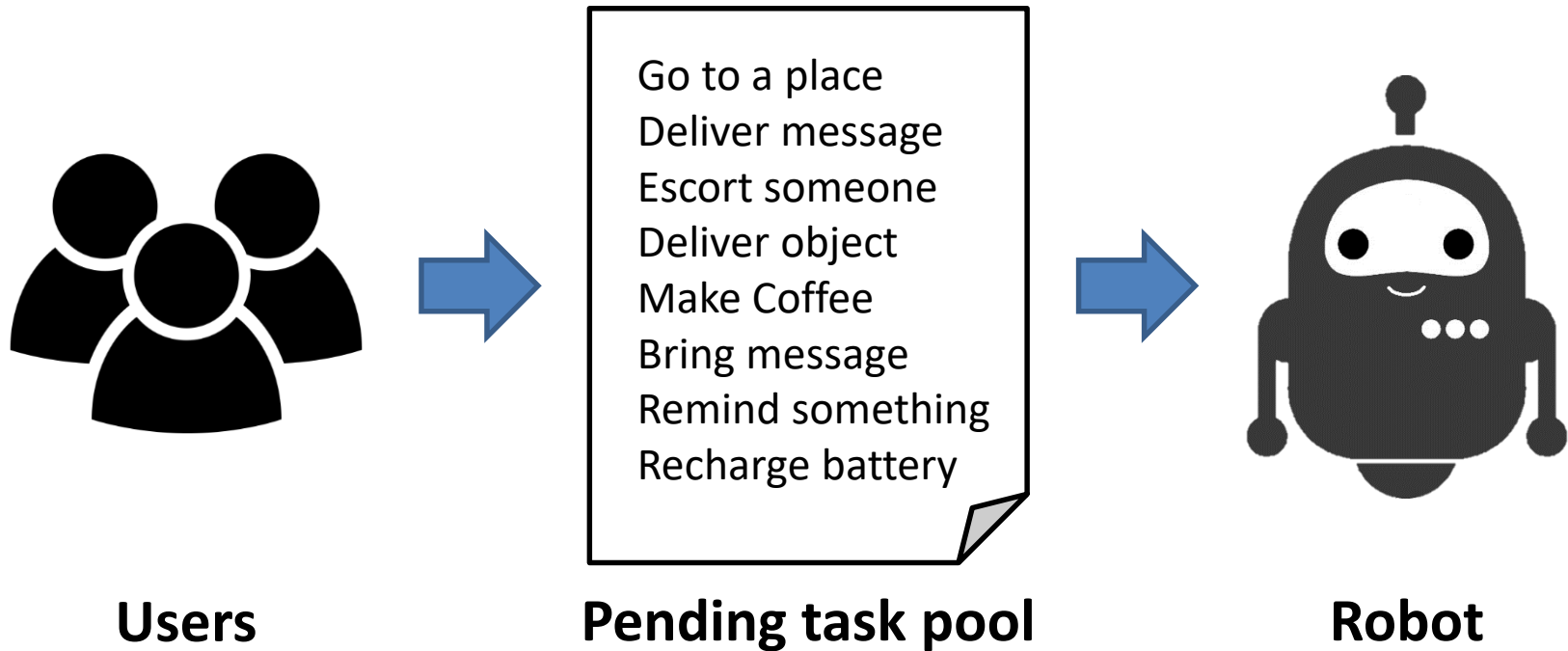
*José Carlos González, Manuela Veloso,*

*Fernando Fernández and Ángel García-Olaya*

Planning and Learning Group

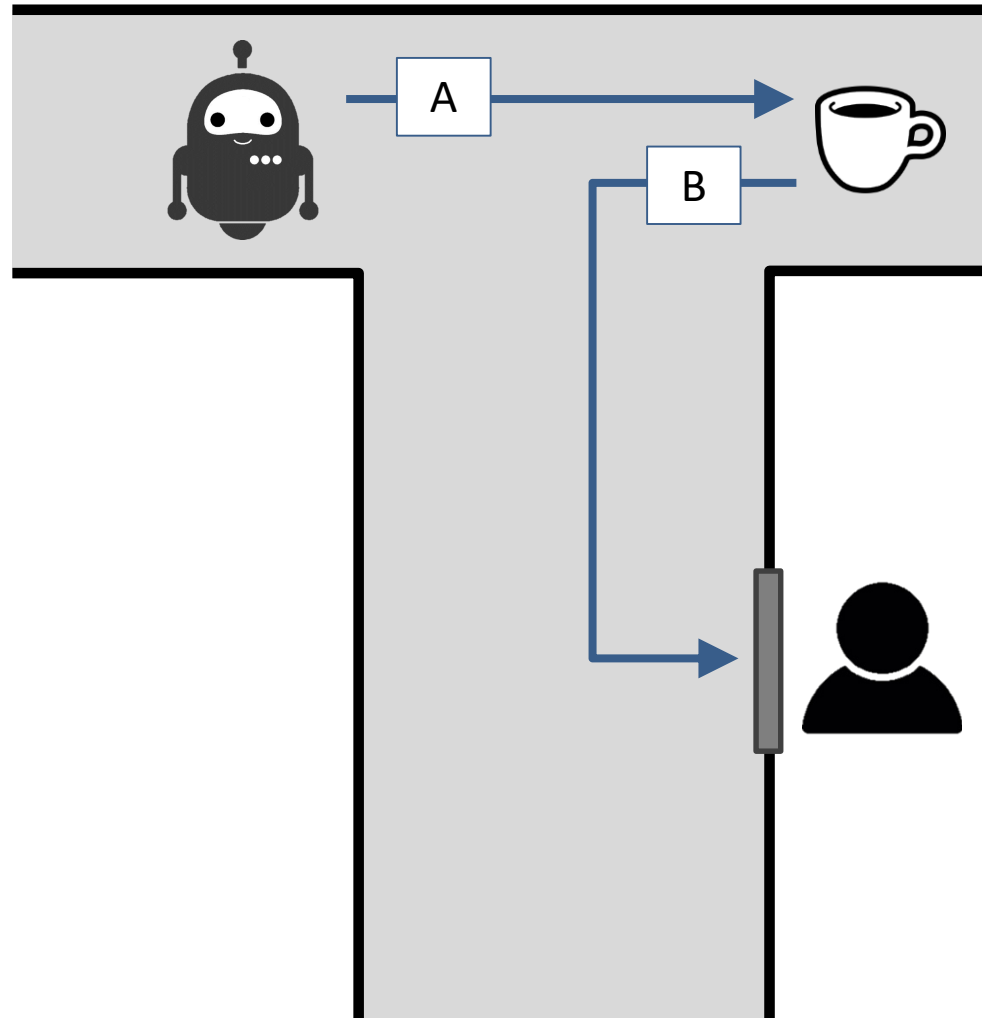


- Robot must find a valid task schedule, and execute it
- Several constraints per task
- Users can add tasks anytime



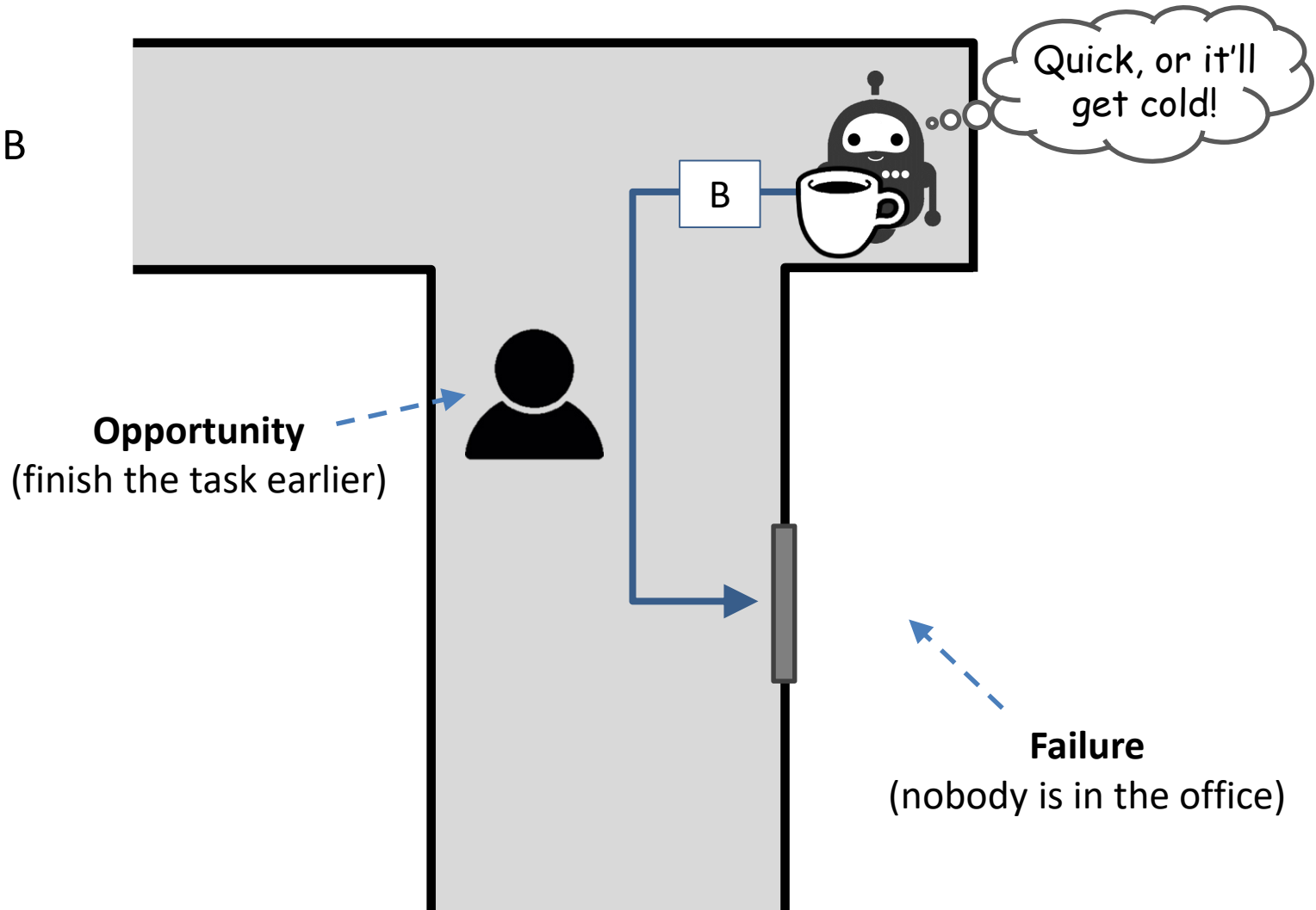
# Introduction – Hot coffee delivering

Subtasks: A, B



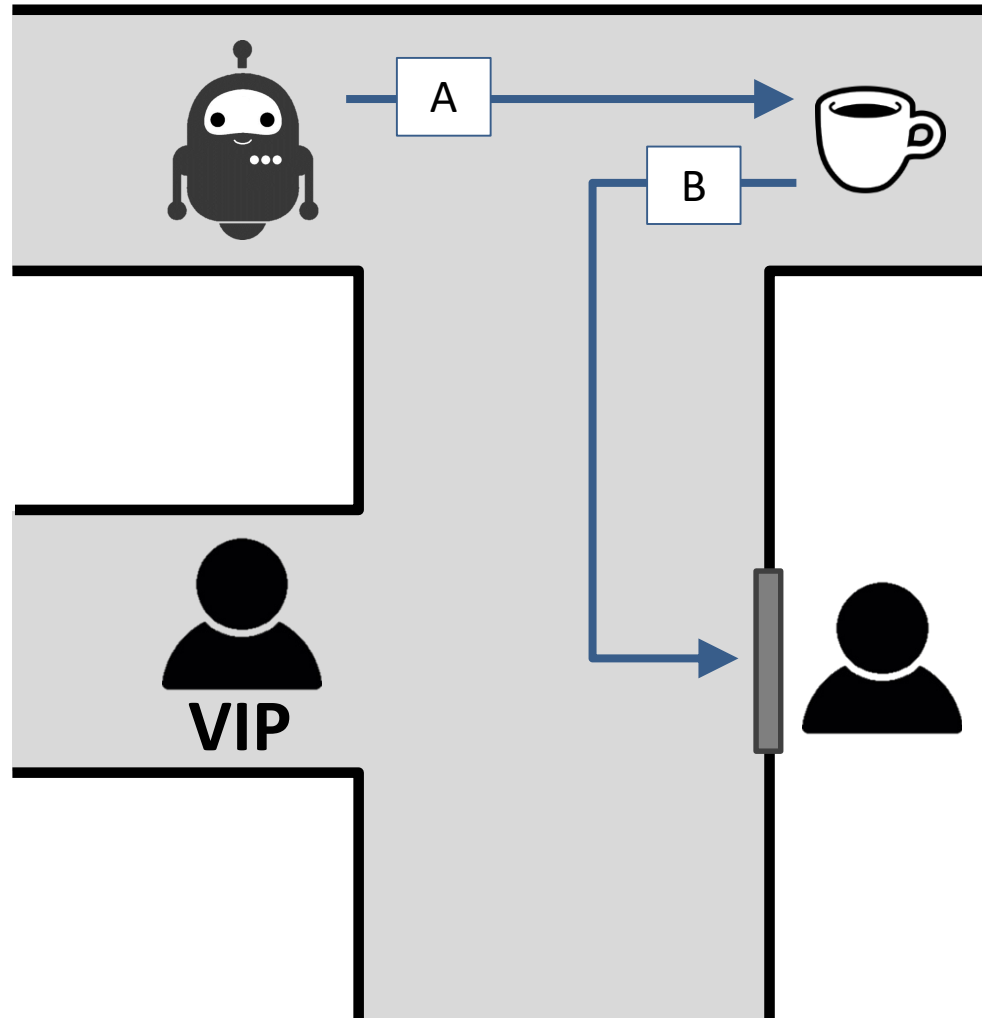
# Introduction – Hot coffee delivering

Subtasks: A, B



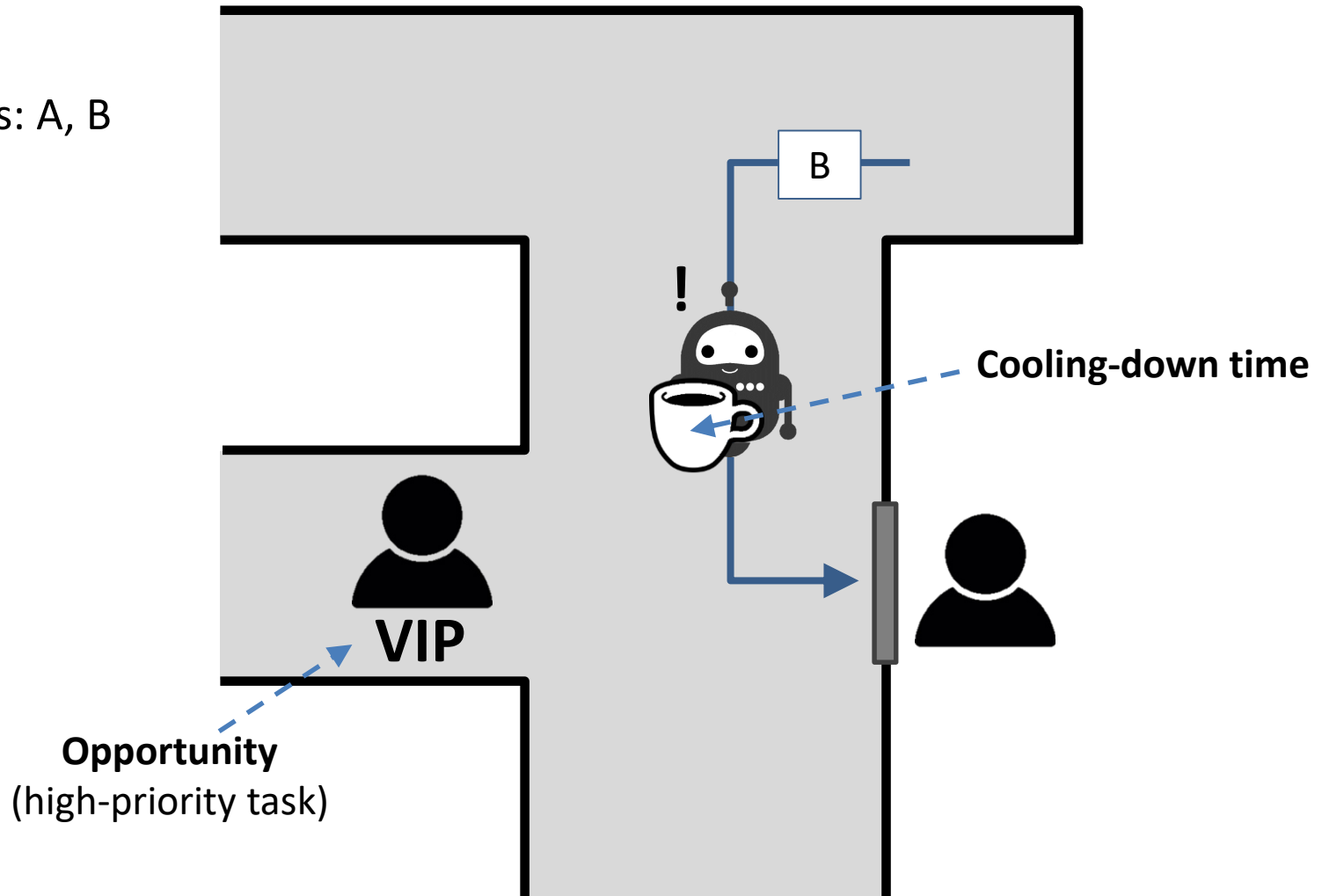
# Introduction – Hot coffee delivering

Subtasks: A, B



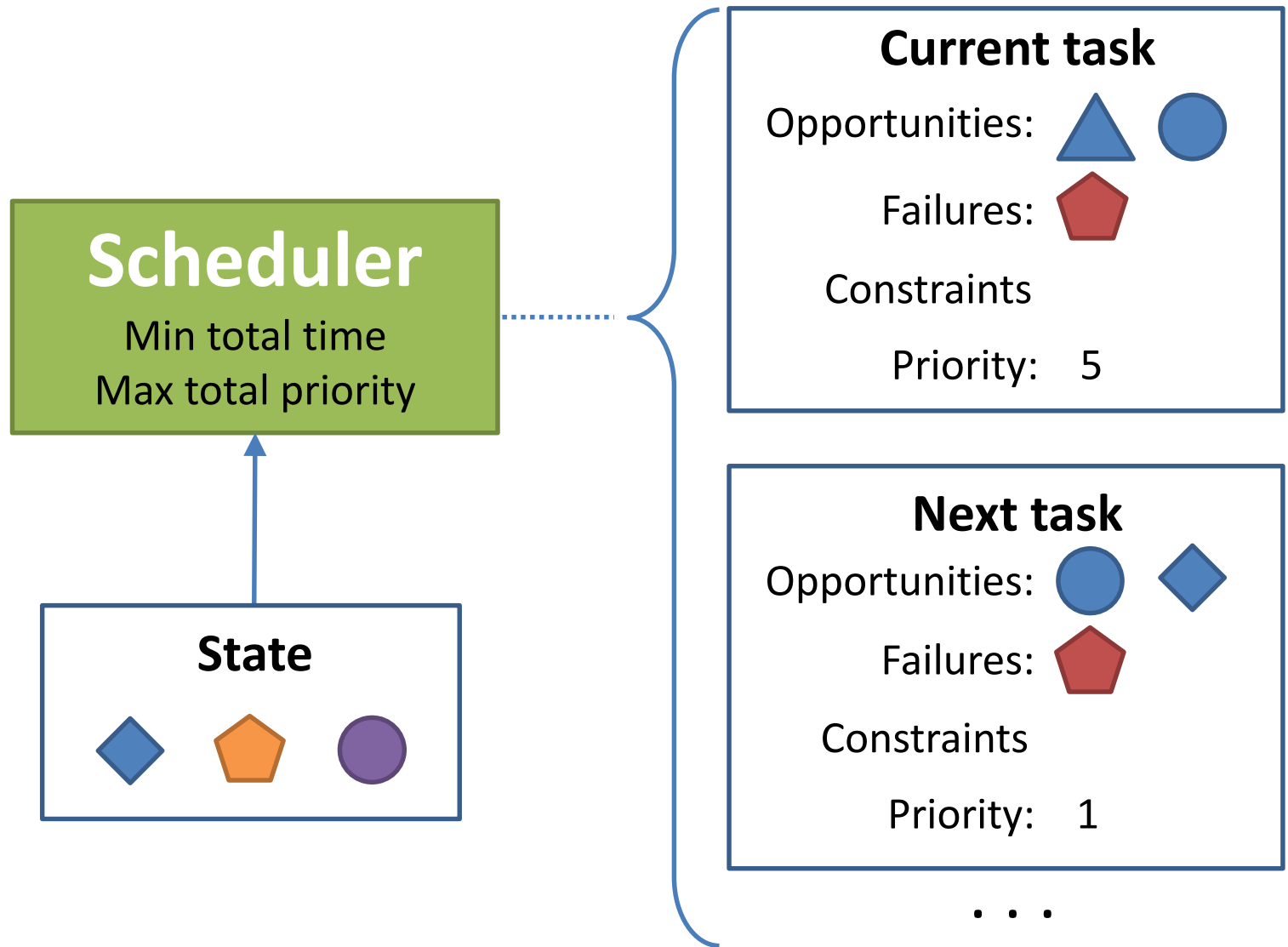
# Introduction – Hot coffee delivering

Subtasks: A, B





- VIP first, then resume B
- Redo A and B
- VIP after B
- Cancel A and B
- Cancel VIP
- Try a quick VIP





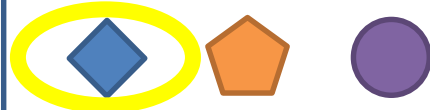
High-level events must be checked  
for all scheduled tasks

## Scheduler

Min total time  
Max total priority

Reschedule!

## State



## Current task

Opportunities:  

Failures: 

Constraints

Priority: 5

## Next task

Opportunities:  

Failures: 


Constraints

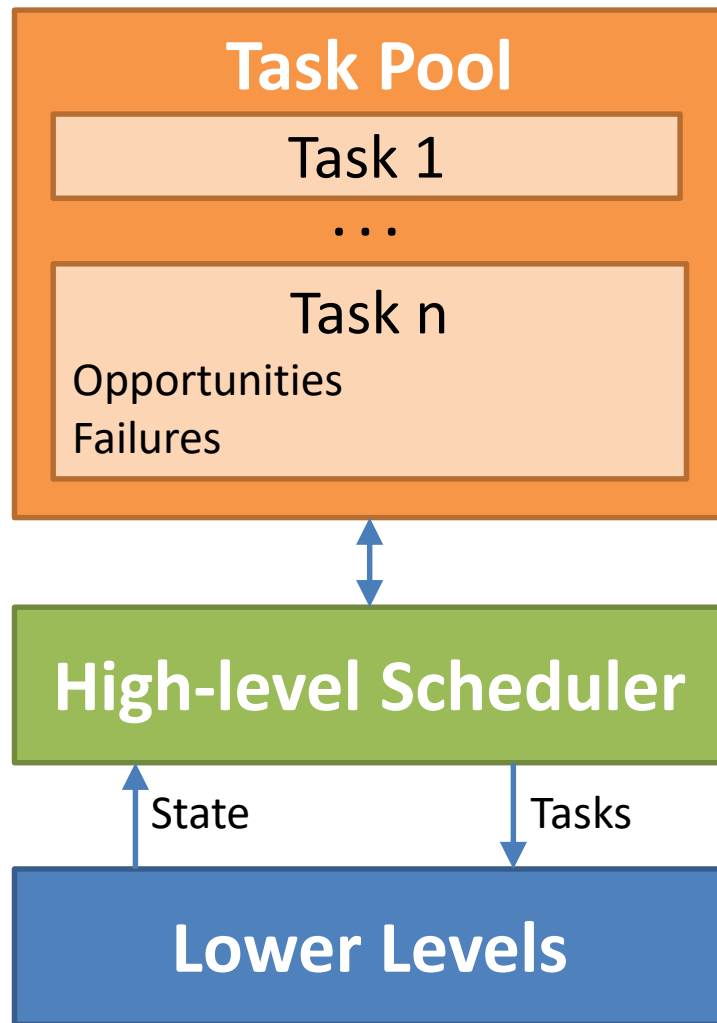
Priority: 1

...

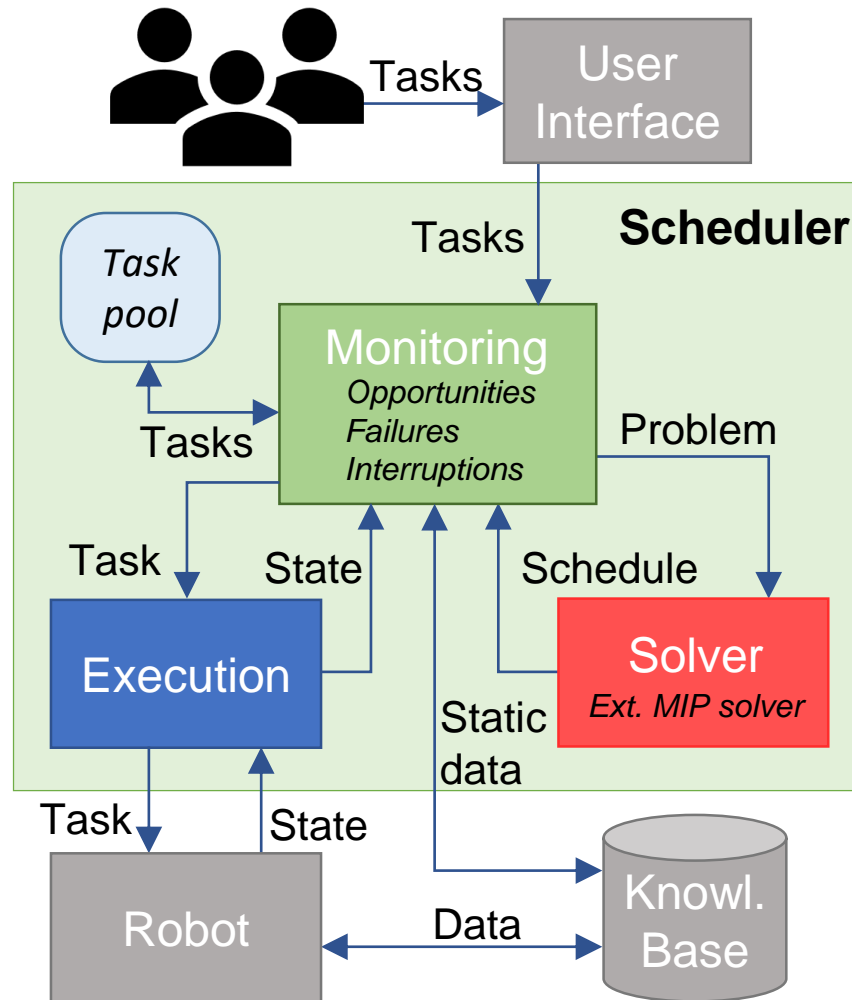
- **Our contribution**

- **Component** to handle high-level unexpected events among tasks
  - MIP model with dependent tasks and **cooling-down times**
- 

- Coltin, B.; Veloso, M. M.; and Ventura, R. 2011.  
Dynamic user task scheduling for mobile robots
    - Fixed schedules with a Mixed Integer Programming (MIP) solver
  - Cashmore, M.; Fox, M.; Long, D.; et al. 2017.  
Opportunistic Planning in Autonomous Underwater Missions
  - Schermerhorn, P.; Benton, J.; Scheutz, M.; et al. 2009.  
Finding and Exploiting Goal Opportunities in Real-Time During Plan Execution
- 
- Our starting point**



- **Updated states** received while subtasks are being executed
- **Generic task attributes**  
*Opportunities* and *Failures*
  - Indicate parameters in the state that should remain invariant
  - Used to trigger reschedulings
- A **rescheduling** can
  - Add or remove tasks in the pool
  - Interrupt the current subtask



- **Multilevel global scheme**

- Rescheduling for high-level events
- Tasks sent to lower abstraction levels
- States are generalized from lower levels

	Task	Subtask-1	Subtask-2
<b>User</b>	Task type	DeliverDrink	MakeHotDrink
	Task owner	Alice	Alice
	Location start	-	CoffeMaker
	Location end	-	CoffeMaker
	Time start min	0	0
	Time end max	15	15
	Person target	Alice	Alice
	Object	HotCoffee	HotCoffee
	Priority	-	10
<b>Internal</b>	Time operation	-	5
	Time cooldown	-	6
	Task depending	-	Subtask-1
	Opportunities	VIP	Person target, VIP
	Failures	TO, BP	HotCoffee, TO, BP

**Constraints:**

$$w_i^{min} \leq s_i \leq w_i^{max} - o_i - d(l_i^s, l_i^e)$$

$$w_i^{min} + o_i + d(l_i^s, l_i^e) \leq e_i \leq w_i^{max}$$

$$Previous(i, j) \Rightarrow s_i < e_i < s_j$$

$$\neg Previous(i, j) \Rightarrow s_i < s_k < s_j$$

$$Previous(i, j) \Rightarrow e_j \geq s_j + o_j + d(l_i^e, l_j^s) + d(l_j^s, l_j^e)$$

$$Depends(j, i) \Rightarrow e_i < s_j$$

$$Depends(j, i) \Rightarrow c_j \geq e_j - e_i$$



Order and overlapping



Depending subtasks  
and cooling-down

**Objective function:**

$$\text{Minimize } \sum_{i=1}^n e_i p_i$$

**Checks:**

$$w_i^{min} + o_i + d(l_i^s, l_i^e) < w_i^{max}$$

$$c_i \geq o_i + d(l_i^s, l_i^e)$$

**Solution types**

- Proven optimal
- Suboptimal
- Not found
  - Unfeasible
  - Time limit

**Positive integer parameters:**

$i, j, k$ : Any task of the pool

$w^{min}$ : Minimum start time

$w^{max}$ : Maximum end time

$s$ : Start time (variable)

$e$ : Ending time (variable)

$o$ : Operation time

$c$ : Cooling down time

$p$ : Priority value higher than 0

$l^s$ : Starting location

$l^e$ : Ending location

$d(a, b)$ : Distance (time estimation) between  $a$  and  $b$

**Binary parameters:**

$Previous(i, j)$ : Task  $i$  starts just before  $j$  (variable)

$Depends(j, i)$ : Task  $j$  must start after  $i$

- If the scheduler **cannot find a suitable plan**
  - **Failures:** Monitoring cancels the next task
    - With the lowest priority first
    - Then the smallest time window that overlaps another
  - **Opportunities:**
    1. Tries to redo the current subtask later
    2. If it cannot, it tries to redo the whole task
    3. If it cannot, it evaluates whether to cancel the current task or the new task by maximizing the gain measure  $g$

**Gain:**  $g = \sum_{i=1}^n p_i$

Sum of the priorities of the scheduled tasks



- **Using the CoBot platform**
  - Their task catalog
  - Schedules work in the actual robot
- **180 simulations**
  - Scheduling times
  - Quality



- **Task decomposition** allows to optimize locations

Schedule 1

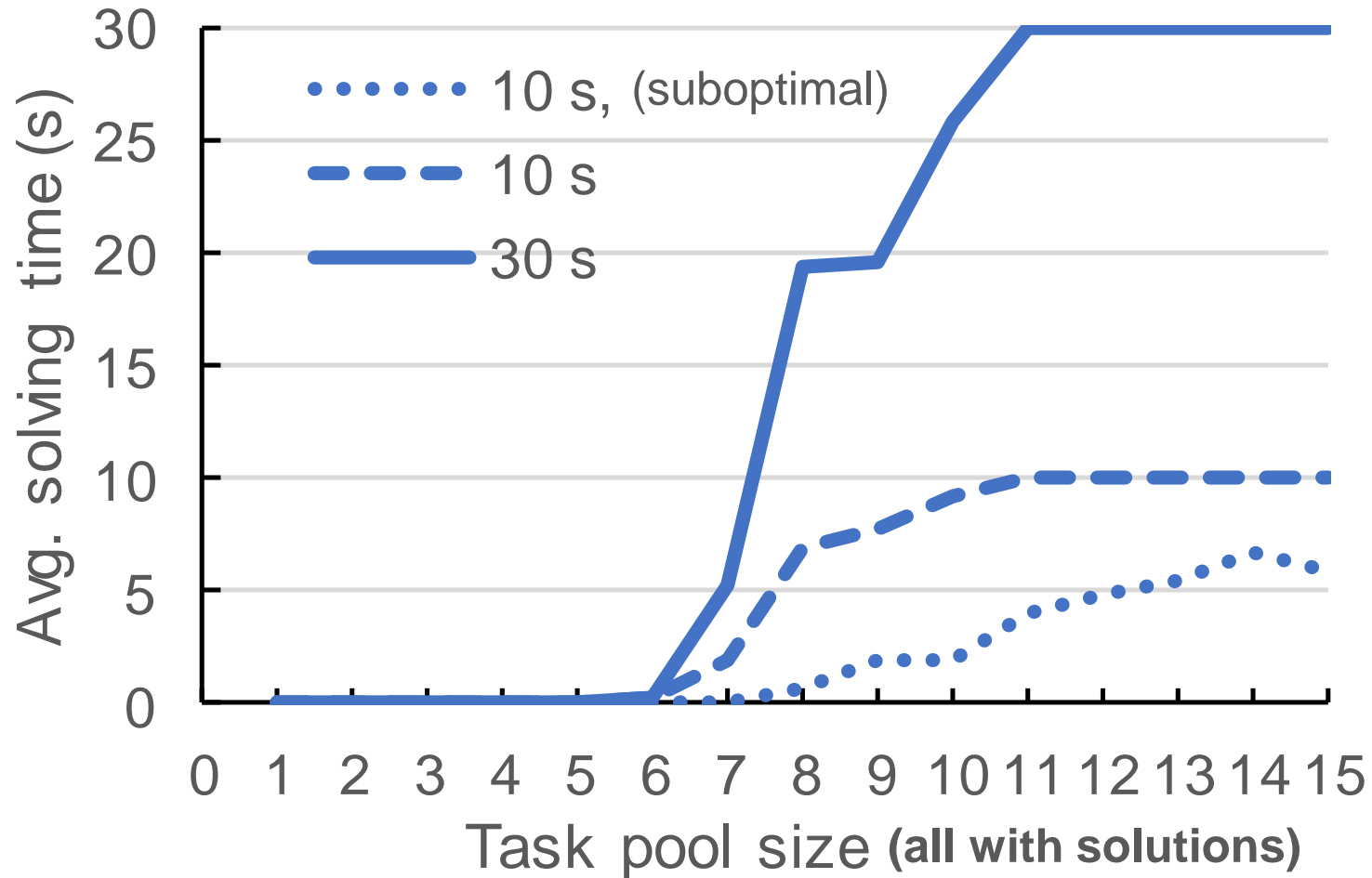
Task	Start	End
...	0	10
C1a	11	20
C2a	21	26
C1b	27	31
C2b	32	33
C3a	34	42
C3b	43	47
VIP	48	53
<b>Cost</b>	739	

Schedule 2

Task	Start	End
...	0	10
C1a	11	20
C2a	21	26
C1b	27	31
C2b	32	33
VIP	34	39
C3a	40	45
C3b	46	50
<b>Cost</b>	605	

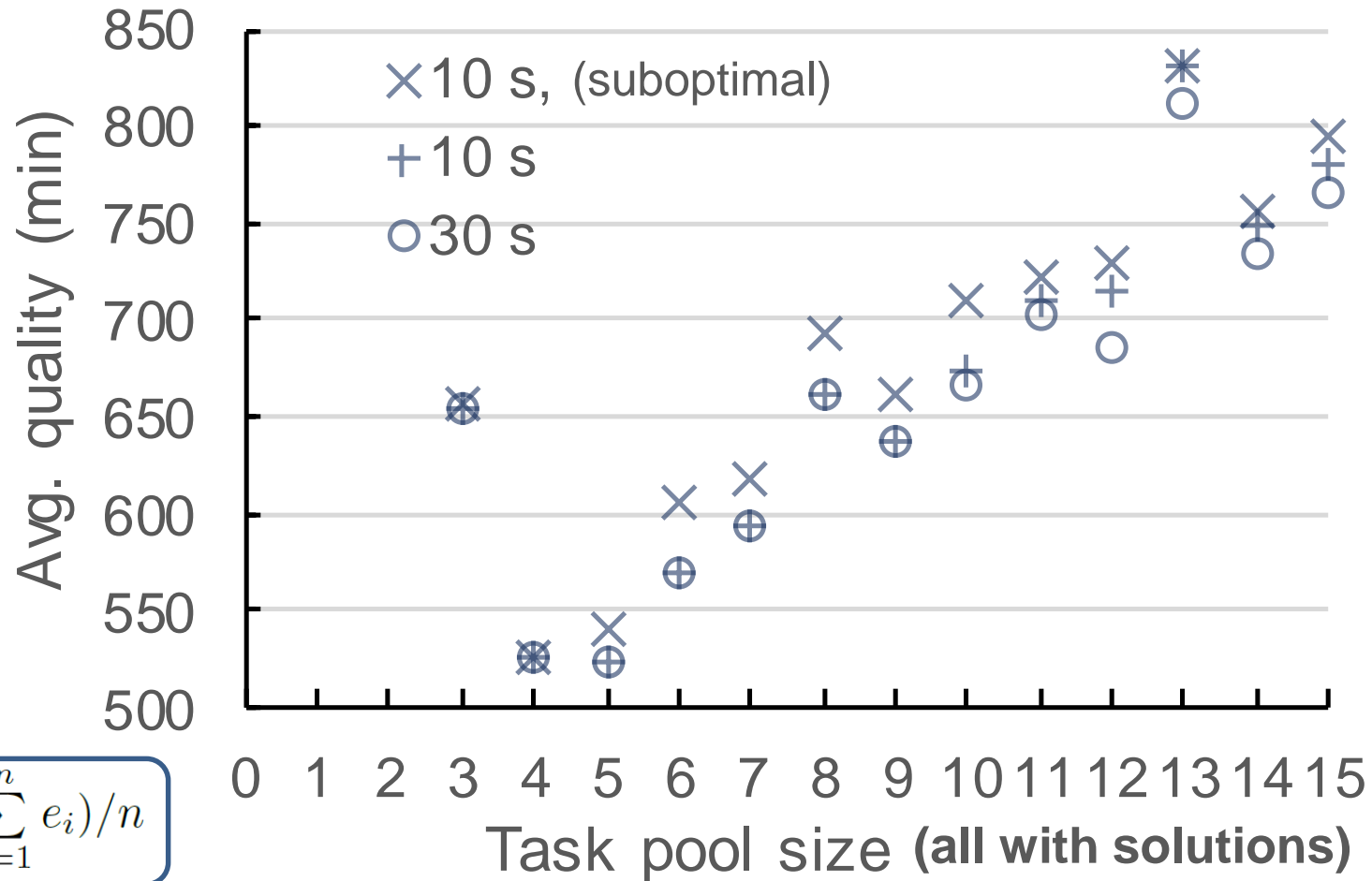
Schedule 3

Task	Start	End
...	0	10
C1a	11	20
VIP	21	23
C2a	24	29
C1b	30	34
C2b	35	36
C3a	37	45
C3b	46	50
<b>Cost</b>	454	



- Proven optimal solutions found up to size 10

## Experiments – Quality vs. Subtasks



$$q = (\sum_{i=1}^n e_i) / n$$

- Quality in “10s suboptimal” is acceptable for the CoBot’s domain

- **New architecture** of task execution, monitoring and rescheduling
  - **Rescheduling** according to **opportunities and failures**
  - **Interruption of tasks** in the middle of their execution
  - **Future work:** integration with a generic hierarchical control architecture, independent from the planning/scheduling mechanism
- **Improved MIP model**
  - Able to deal with **cooling-down times** and dependent tasks
  - Focused on the **quality of the solutions**
  - **Quality can be affected in extreme conditions** with large task pools and fast solving times required
  - **Future work:**
    - Transform some hard-constraints (time-window) into soft
    - Comparisons with other rescheduling systems

# Task Monitoring and Rescheduling for Opportunity and Failure Management

*José Carlos González, Manuela Veloso,*

*Fernando Fernández and Ángel García-Olaya*

Planning and Learning Group

# Thank you for your attention

- **High-level events**
  - Affect the **current task and future tasks** in the schedule
  - **Interrupt tasks** in the middle of their execution
- **Opportunities**
  - Domain: can appear at any moment (VIP)
  - Specific: exclusive for a task (receipt of the coffee found earlier)
- **Failures**
  - Domain: same failure for several tasks (blocked paths, timeout)
  - Specific: exclusive for a task (coffee stolen)



## Experimental sets $A > B > C$

- **A:** 480 random instances (task pools)
- **B:** 12 solved instances per each pool size from 1-15 (180 in total)
- **C:** 12 random instances per each pool size from 8-15 (96 in total)

## Experiments – Solution types

Configuration		10 s, 4.4% tol.	10 s	30 s
Set A	Time out: no solut.	11.0%	11.0%	8.8%
	Proven unfeasible	0.8%	0.8%	0.8%
	Check failed	4.4%	4.4%	4.4%
	Proven optimal	16.3%	42.7%	43.1%
	Min. gap reached	54.0%	0.0%	0.0%
	Time out: found	13.5%	41.0%	42.9%
	Solutions found	83.8%	83.8%	86.0%
Set B	Proven optimal	17.8%	51.1%	52.2%
	Min. gap reached	68.3%	0.0%	0.0%
	Time out: found	13.9%	48.9%	47.8%
	Av. solver time (s)	2.14 ± 3.6	5.07 ± 4.9	14.7 ± 14.8
	Av. quality (min)	611 ± 256	596 ± 250	590 ± 247
Set C	Proven optimal	0.0%	10.4%	12.5%
	Min. gap reached	74.0%	0.0%	0.0%
	Time out: found	26.0%	89.6%	87.5%
	Av. solver time (s)	3.98 ± 4.1	9.24 ± 2.5	26.88 ± 8.6
	Av. quality (min)	738 ± 135	721 ± 137	709 ± 137