simula

Constraint-guided Test Execution Scheduling: An Experience Report at ABB Robotics

Arnaud Gotlieb[1], Morten Mossige[2], Helge Spieker[1]

Safecomp 2023, Toulouse, France
Sep. 20, 2023

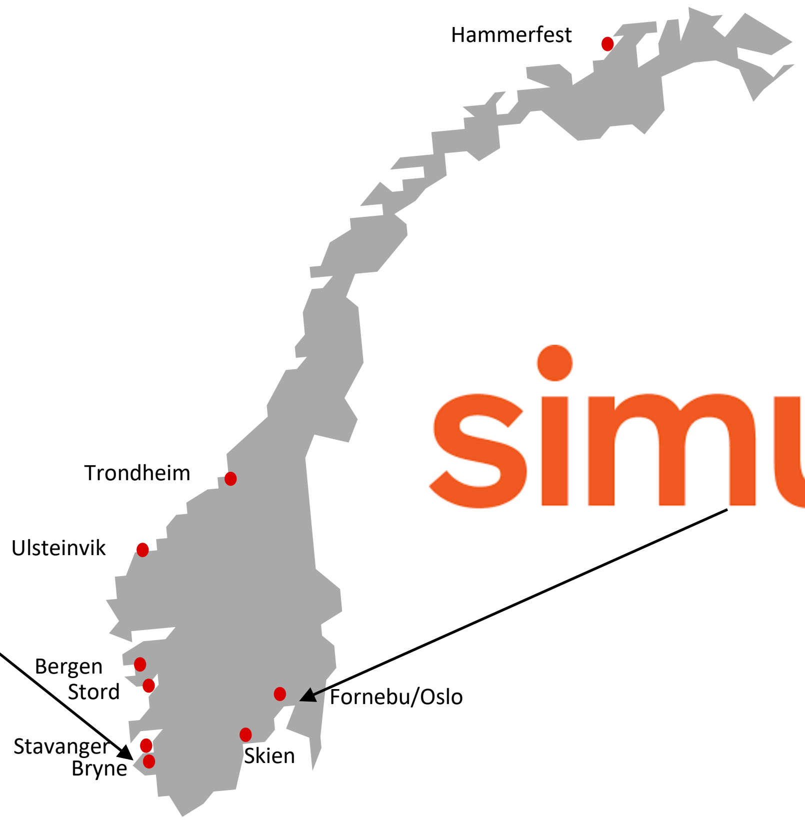
1. Simula Research Laboratory, Oslo, Norway
2. ABB Robotics, Bryne, Norway

ABB in Norway - overview

Key information

- Turnover: 8.4 Billion NOK, No. of Employees: 1944
- Lead business: Energy Industries
  - Electrification
  - E-mobility
  - Marine and Ports
  - Motion
  - Robotics and Discrete Automation

ABB ROBOTICS

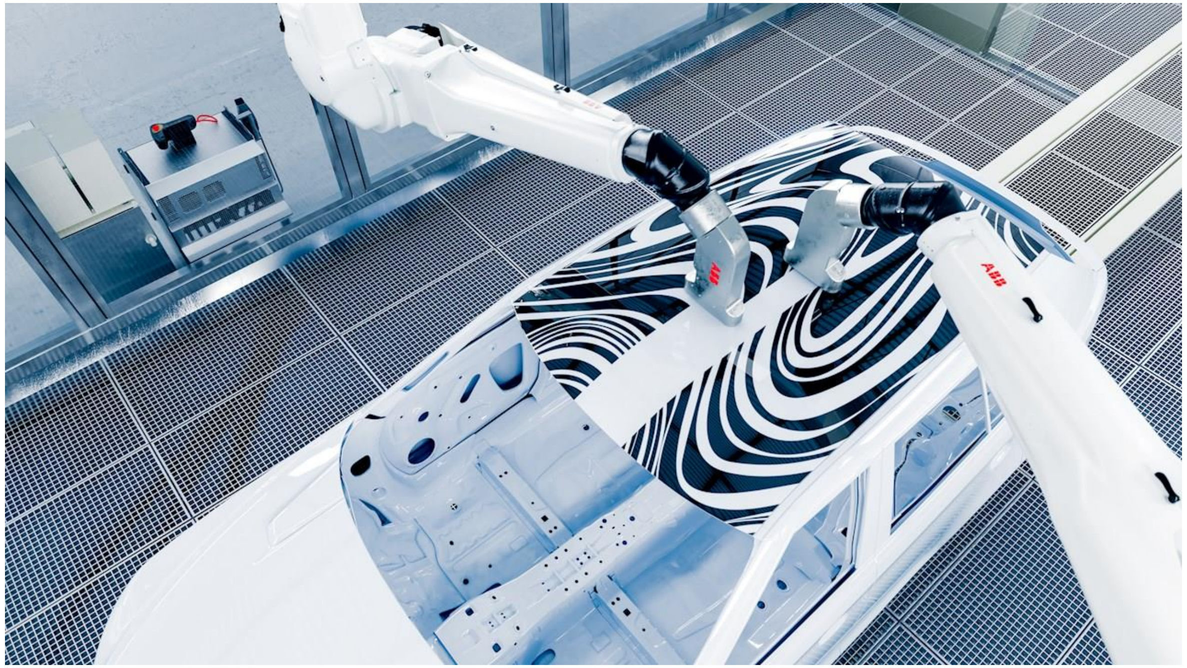


simula

ABB

ABB Robotics (and Discrete Automation), Norway

- World-class R&D for paint robots, Bryne
- Competence/development crucial for new ABB products
- World's first paint robot in 1969, sold to Gustavberg i Sweden
- RobotNorge handles all ABB industrial robot sale, engineering, and service in Norway



World-class R&D for paint robots

Robots Product Line Testing

PRODUCT	BASIC SPECIFICATIONS	
IRB 14000 YuMi® 	Load (kg)	0.50
	Reach (m)	0.559
	Protection	Std:IP30; Clean room ISO 5
	Mounting	Bench, table
	Safety	PL b Cat B
IRB 14050 Single Arm YuMi 	Load (kg)	0.50
	Reach (m)	0.559
	Protection	Std:IP30; Clean room ISO 5
	Mounting	Any angle - table, wall, ceiling
	Safety	PL d Cat 3, PL b Cat b, SafeMove Pro option
IRB 1100 	Load (kg)	4.00 4.00
	Reach (m)	0.475 0.58
	Armload (kg)	0.50 0.50
	Protection	Std: IP40
	Mounting	Any angle
IRB 120 and IRB 120T 	Load (kg)	3.00
	Reach (m)	0.58
	Protection	Std: IP30 Option: Cleanroom class 5, certified by IPA
	Mounting	Floor, wall, inverted, and tilted angles

PRODUCT	BASIC SPECIFICATIONS	
IRB 1200 	Load (kg)	5.00 7.00
	Reach (m)	0.90 0.70
	Protection	Std: IP40 Option: IP67, Clean room ISO 4, food grade lubricant
	Mounting	Any angle
IRB 140 and IRB 140T 	Load (kg)	6.00
	Reach (m)	0.81
	Protection	Std: IP67 Option: Cleanroom class 6, Foundry Plus
	Mounting	Floor, wall, inverted, and tilted angles
IRB 1600 	Load (kg)	6.00 6.00 10.0 10.0
	Reach (m)	1.20 1.45 1.20 1.45
	Protection	Std: IP54 Option: IP67 with foundry plus 2
	Mounting	Floor, wall, inverted, tilted angles, and shelf
IRB 1660ID 	Load (kg)	4.00 6.00
	Reach (m)	1.55 1.55
	Protection	Std: IP40 (wrist IP67)
	Mounting	Floor, wall, inverted, and tilted angles

From a concrete set up:

<u>Test Case Repository:</u>
~10,000 Test Cases (TC)
~25 distinct Test Robots
~500 distinct features

10..30 code changes per day

→ Select, schedule and execute about 150 TC per Continuous Integration cycle

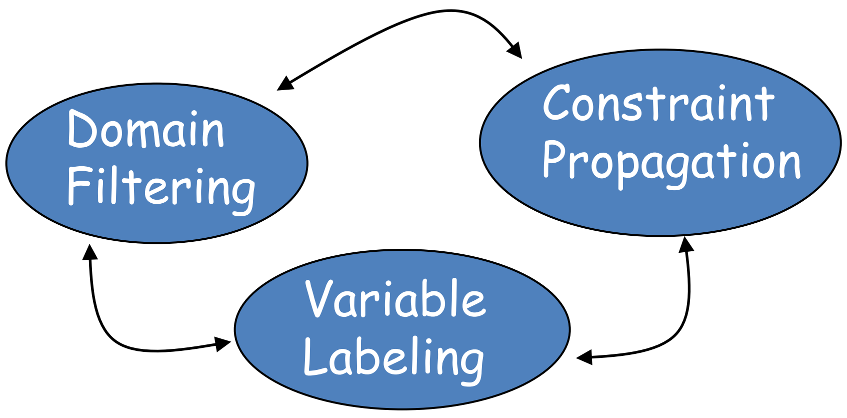
simula

Problem to Solve

How to schedule the execution of a maximum of test cases, over all the available robots, during each CI cycle?

- A global optimization problem!
- Global resources to be shared (oscilloscope, paint conveyor, etc.)
- With sufficient diversity in the testing process
- Solving the problem is time-constrained!

Artificial Intelligence/Constraint Programming (CP)



- Routinely used in Validation & Verification,
  CP handles efficiently hundreds of thousands
  of constraints and variables

- CP is versatile: user-defined constraints, dedicated solvers, programming search heuristics **but it is not a silver bullet**
  (developing efficient CP models and heuristics requires expertise)

→ **Global constraints**: relations over a non-fixed number of variables, implementing dedicated filtering algorithms

The **nvalue** global constraint

[Pachet Roy 1999, Beldiceanu 01]

nvalue(N, V)

Where:

N is a finite-domain variable

$V = [V_1, \dots, V_k]$ is a vector of variables

nvalue(N, V) holds iff $N = card(\{V_i\}_{i \ in \ 1..k})$

nvalue(N, [3, 1, 1, 3, 3, 3, 1, 1, 1]) entails N = 2

nvalue(3, $[X_1, X_2]$) fails

nvalue(1, $[X_1, X_2, X_3]$) entails $X_1 = X_2 = X_3$

N in 1..2, **nvalue**(N, [4, 7, X_3]) entails X_3 in {4,7}, N=2

Has been used successfully in Test Suite Reduction Problem!

A. Gotlieb and D. Marijan. ***Using Global Constraints to Automate Regression Testing***. AI Magazine 38, no. Spring (2017).

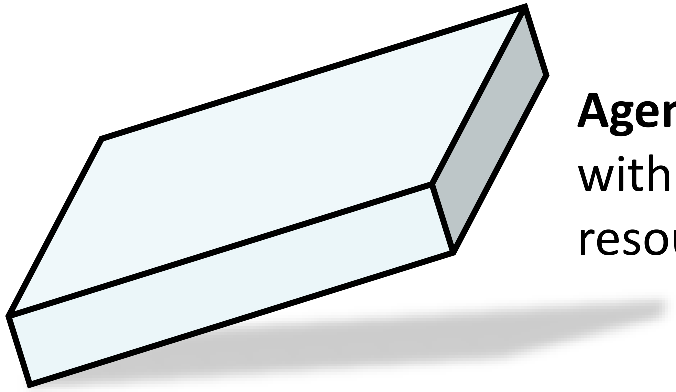
Constraint-Based Scheduling

<u>Assignment of **Tasks to Agents** such that:</u>

1. Task execution is not interrupted or paused
2. Agents are maximally occupied
3. Tasks sharing a global resource cannot be executed at the same time
4. Diversity of assignment of tasks to agents is ensured

Tasks
with distinct characteristics

Schedule



Agents
with limited time or resources capacity

<u>Goal:</u>
Schedule as much tasks as possible on available agents such that the overall execution time is minimized

The CUMULATIVE global constraint [Aggoun & Beldiceanu AAAI'93]

CUMULATIVE(t, d, r, m)

Where

$t = (t_1, \ldots, t_N)$ is a vector of tasks, each t_i in $S_i \,..\, E_i$

$d = (d_1, \ldots., d_N)$ is a vector of task duration

$r = (r_1, \ldots, r_N)$ is a vector of resource consumption rates

m is a scalar

CUMULATIVE ***(t, d, r, m)*** holds iff

$$\sum_{\substack{i=1 \\ t_i \leq t \leq t_i + d_i}}^{N} r_i \leq m$$

Test Case Execution Scheduling

(T, M, G, d, g, f)

***T**: a set of Test Cases*
***M**: a set of Machines, e.g., robots*
***G**: a set of (non-shareable) resources*

$d: T \rightarrow N$ *estimated duration*
$g: T \rightarrow 2^G$ *usage of global resources*
$f: T \rightarrow 2^M$ *possible machines*

Disjunctive scheduling,
non-preemptive,
non-shareable resources,
machine-independant
execution time

Function to optimize:
TimeSpan: the overall duration of test execution T_E
(in order to minimize the round-trip time, i.e., time required to execute all the test cases)

In practice, global optimality is desired but not mandatory, it's more important to control the time to compute the schedule → Time-constrained global optimization (Good enough solution!)

A simple example

	d	*f*	*g*
Test	Duration	Executable on	Use of global resource
$t1$	2	$m1, m2, m3$	-
$t2$	4	$m1, m2, m3$	$r1$
$t3$	3	$m1, m2, m3$	$r1$
$t4$	4	$m1, m2, m3$	$r1$
$t5$	3	$m1, m2, m3$	-
$t6$	2	$m1, m2, m3$	-
$t7$	1	$m1$	-
$t8$	2	$m2$	-
$t9$	3	$m3$	-
$t10$	5	$m1, m3$	-

Test Cases: *t1, t2, t3, t4, t5, t6, t7, t8, t9, t9, t10*

r1

m3

m2

m1

Using the global constraint **CUMULATIVE**

CUMULATIVE$((t_1,..,t_{10}), (d_1,..,d_{10}), (1, ..,1), 3)$,
$M_1,..,M_6$ *in* $1..3$,
$M_7 = 1, M_8 = 2, M_9 = 3, M_{10}$ *in* $\{1,3\}$,
$(E_2 \leq S_3$ *or* $E_3 \leq S_2), (E_2 \leq S_4$ *or* $E_4 \leq S_2)$,
$(E_3 \leq S_4$ *or* $E_4 \leq S_3)$,
MAX$(MaxSpan, (E_1, \ldots, E_{10}))$,
LABEL(MINIMIZE$(MaxSpan), (S_1,..,S_{10}), (M_1,..,M_{10}))$

Test	Duration	Executable on	Use of global resource
$t1$	2	$m1, m2, m3$	-
$t2$	4	$m1, m2, m3$	$r1$
$t3$	3	$m1, m2, m3$	$r1$
$t4$	4	$m1, m2, m3$	$r1$
$t5$	3	$m1, m2, m3$	-
$t6$	2	$m1, m2, m3$	-
$t7$	1	$m1$	-
$t8$	2	$m2$	-
$t9$	3	$m3$	-
$t10$	5	$m1, m3$	-

An optimal solution:

$S_1 = 0, S_2 = 4, S_3 = 8, S_4 = 0, S_5 = 4, S_6 = 7, S_7 = 2, S_8 = 9,$
$S_{10} = 3,$
$M_1 = 1, M_2 = 1, M_3 = 1, M_4 = 2, M_5 = 2, M_6 = 2, M_7 = 1,$
$M_8 = 2, M_9 = 3, M_{10} = 3$
$MaxSpan = 11$

M. Mossige, A. Gotlieb, H. Spieker, H. Meling and M. Carlsson - ***Time-aware Test Case Execution Scheduling for Cyber-Physical Systems*** *- In Proc. of Principles of Constraint Prog. (CP'17), 2017.*

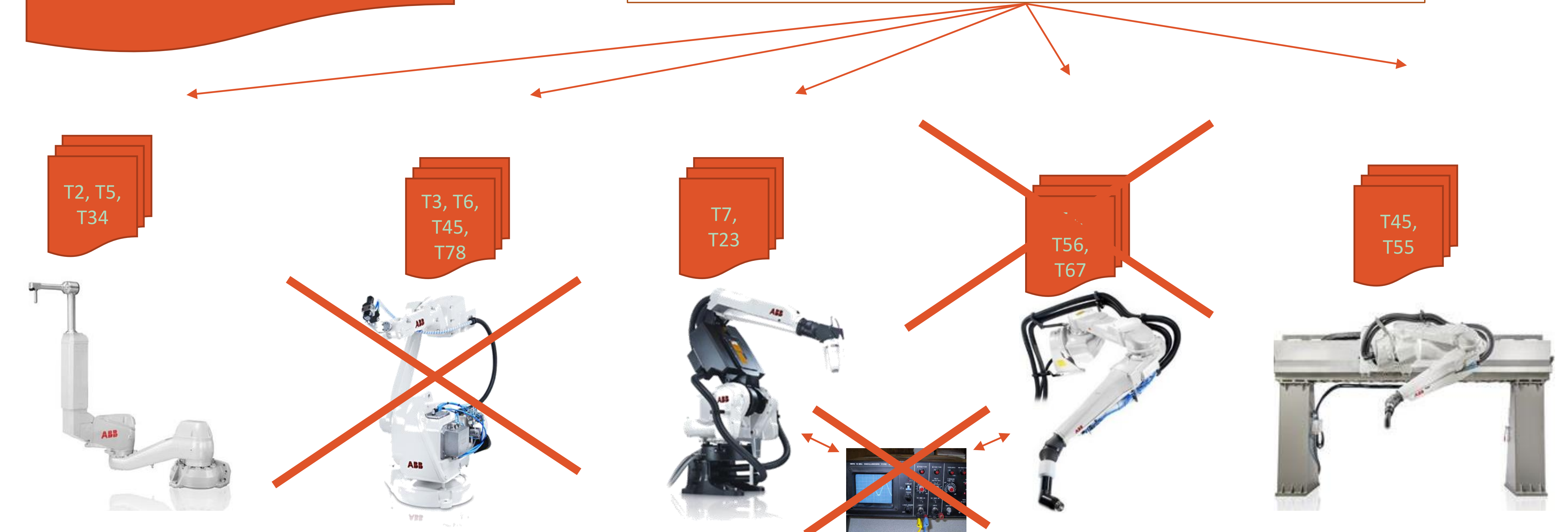
Limitations of this model

- Historical data about test case success/failure is not taken into consideration!
- Diversity in scheduling among CI cycles is not handled
- Static model – In practice, robots and test cases are not necessarily available at each CI cycle → Need a more dynamic model!

A New Approach Based on Multi-Cycles Bin-Packing

A. Test results from n previous runs (Pass/Fail)
B. Developer priority
C. Test duration
D. Time since last execution

- Modeled using the **BIN_PACKING** global constraint
- Computing priorities based on A, B, C (Priority)
- Combined with D (Affinity) with several heuristics
- Incremental solving from CI cycle to CI cycle



simula

The BIN_PACKING global constraint

BIN_PACKING(*items*, *bins*)

Where

items = $(t_1, \ldots, t_N)$ is a vector of *item*, each t_i is $(V_i, size_i)$

bins = $(b_1, \ldots, b_M)$ is a vector of *bin*, each bi is (id_i, C_i)

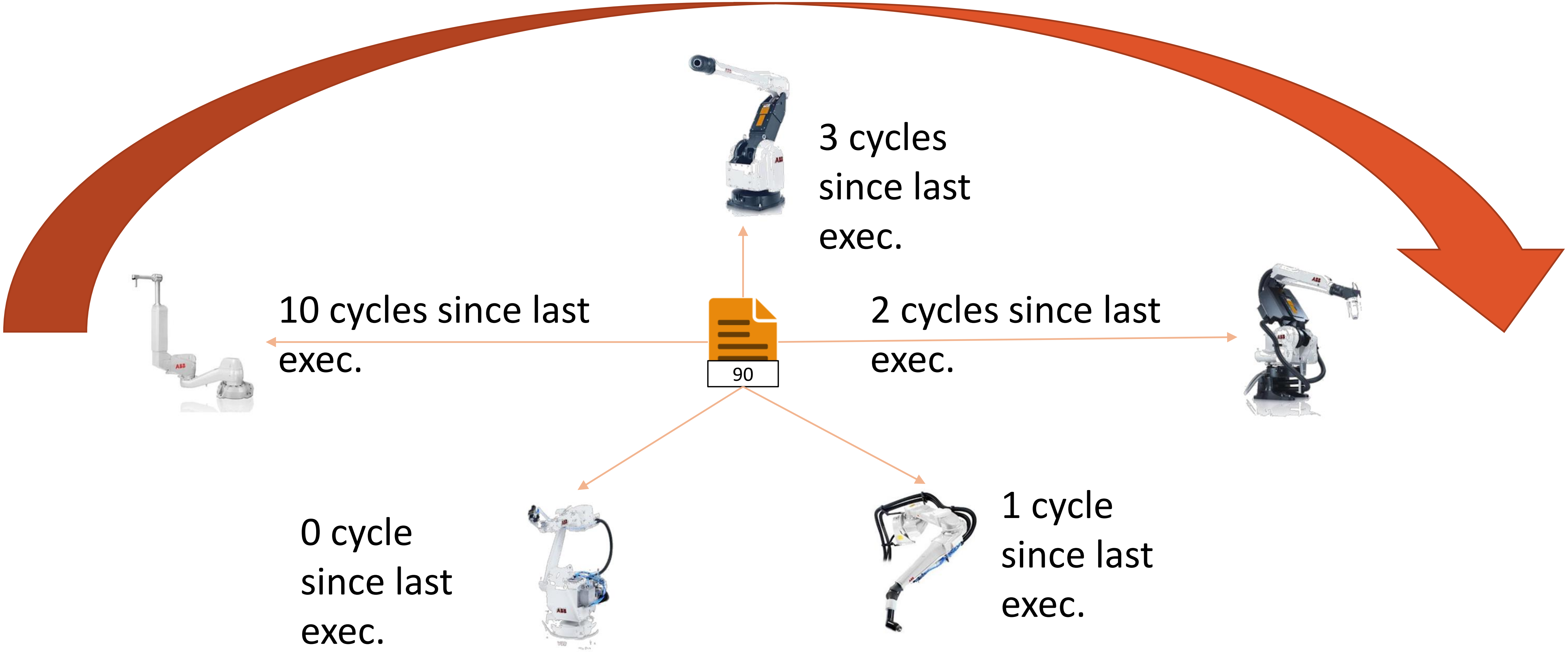
BIN_PACKING *(items, bins)* holds iff every *bin* equals one of the id_i values, and for every bin(id, C_i), the total size of the items assigned to it equals C_i

BIN_PACKING([item(X1, 4), item(X2, 3), item(X3, 5)], [bin(1, Y1), bin(2, Y2)]), Y1 #=< 3, Y2 #=< 11.

entails X1 = 2, X2 = 1, X3 = 2, Y1 = 3, Y2 = 9

Modeled machines as bins and test cases as items → A Very Efficient CP Model to solve the Scheduling Problem!

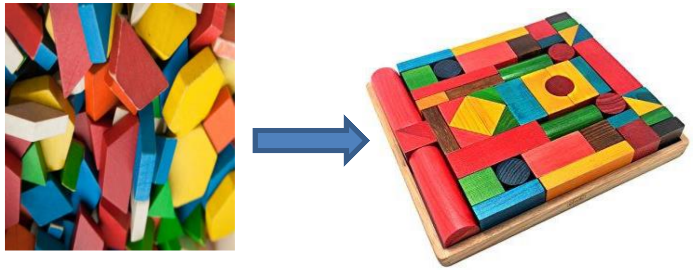
Rotational Diversity: more diversity in the test execution process



H. Spieker, A. Gotlieb and M. Mossige. **Rotational Diversity in Multi-Cycle Assignment Problems**. In Proc. of the AAAI Conference on Artificial Intelligence (AAAI-19). Vol. 33. AAAI, 2019.

SWMOD: Deployment of Time-aware Test Case Execution Scheduling at ABB Robotics

- ~1500 lines of SICStus Prolog Code with CP(FD)
- Fully integrated into the MS-TFS Continuous Integration
- Using the global constraint binpacking + rotational diversity
- Deployed and Continuously Improved at ABB since Feb. 2019



Constraint-based Scheduling

CP with **global constraints (cumulative, binpacking)** and **rotational diversity** can solve the test execution scheduling problem

"SWMOD deployed at ABB Robotics and used every day to schedule tests throughout several ABB centers in the world (Norway, Sweden, India, China)"

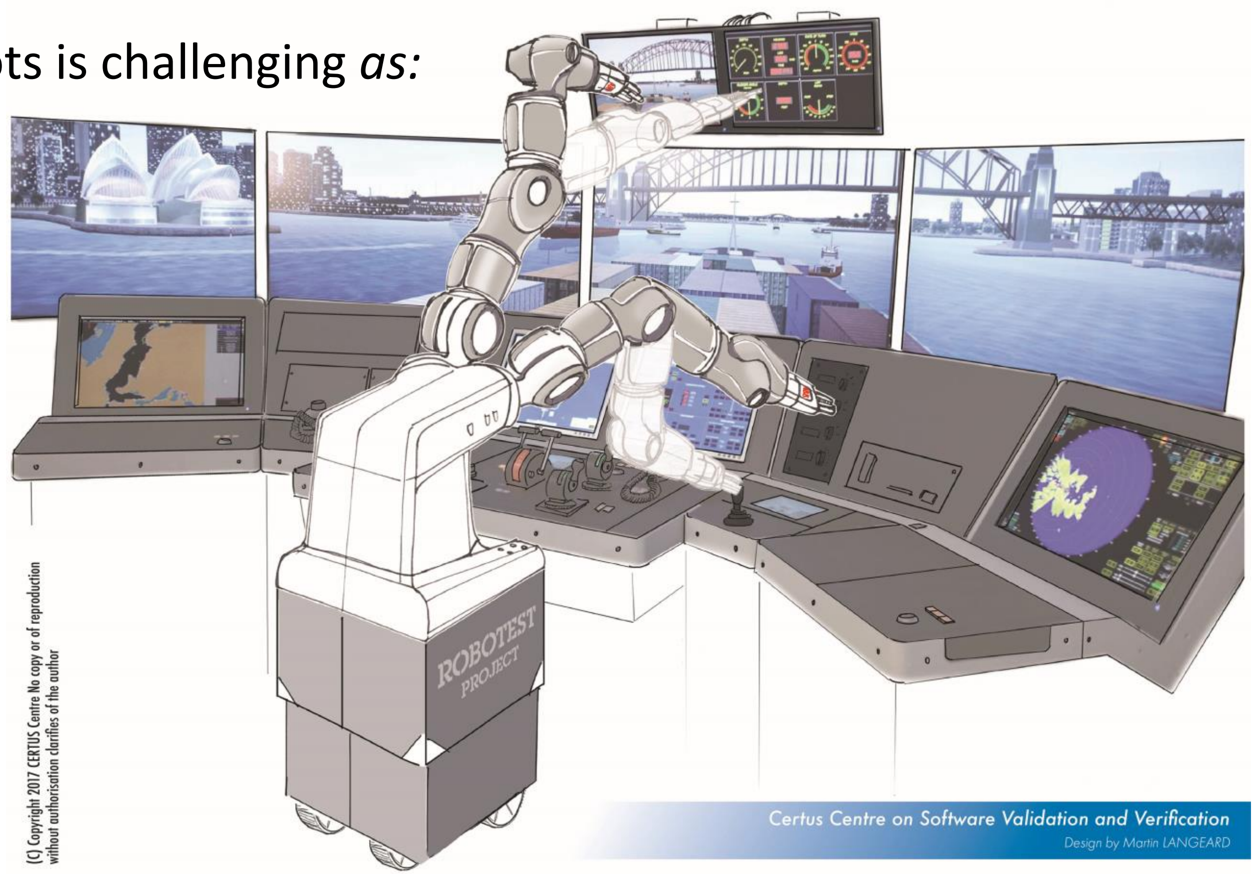
Take Away Message

- *Testing robotics systems* brings new interesting challenges for software V&V research

- **Some AI techniques such as Constraint Programming (CP) and global constraints** are very successful in test case generation, test suite reduction and now test execution scheduling

- Testing autonomous systems such as collaborative robots is challenging *as:*
  - **Expected behaviours** cannot be specified in advance
  - **Interactions with humans** involve more safety issues

We are currently exploring the usage of **Constraint Aquisition** and **Active Learning** methods for testing automated systems



simula

Thanks for your attention!

Constraint-guided Test Execution Scheduling: An Experience Report at ABB Robotics

Arnaud Gotlieb[1], Morten Mossige[2], Helge Spieker[1]