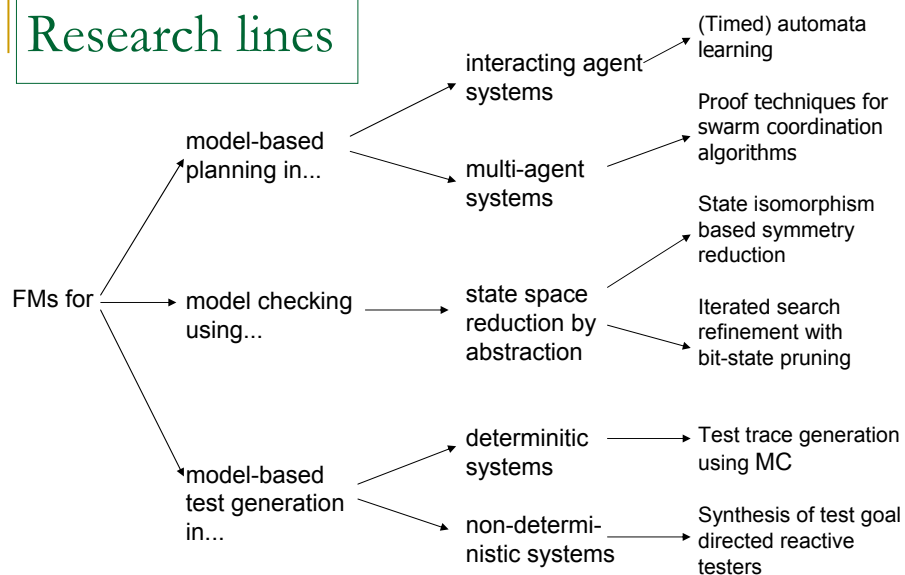


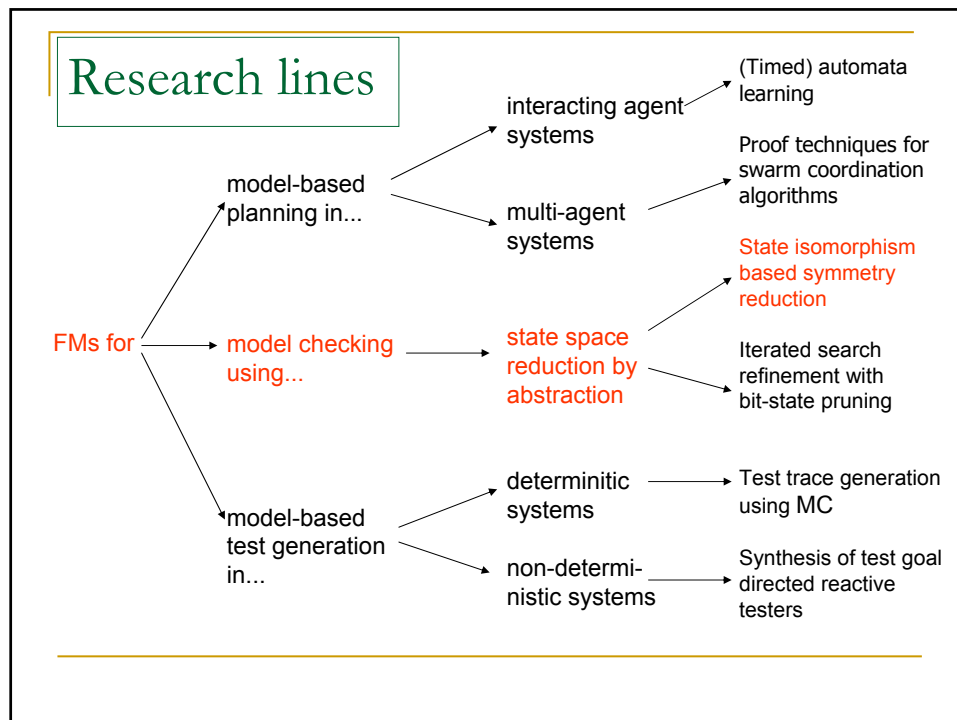
CREDES related research at the Dept. of Computer Science TUT

Jüri Vain

CREDES Kick-off Meeting Tallinn, June 05, 2009

Research lines





State isomorphism based symmetry reduction

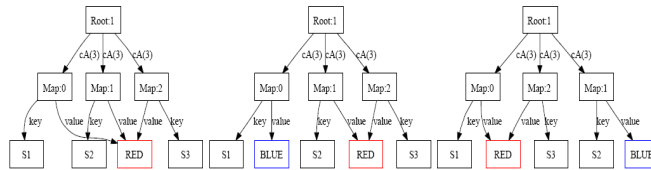
- Solution
 - Given a model in terms of unordered data structures, like **sets** and **maps**, and **objects**:
 - Construct the state graphs of the states at run time and
 - Use the computation of graph isomorphism to determine whether a state with similar structure has been seen.

- Results developed with Margus Veanes and Colin Campell and published at FORTE 2007 (June 2007) and in the PhD thesis of J. Ernits (Nov. 2007)

State isomorphism based symmetry reduction

■ Problem

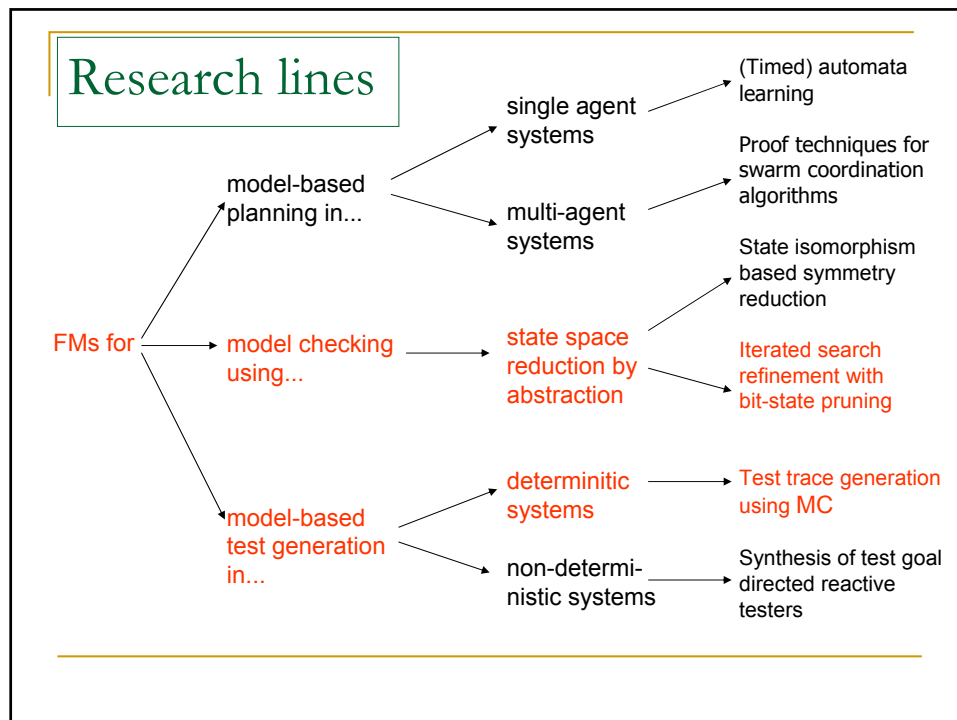
- Exploit symmetries to reduce the effects of state space explosion in explicit state reachability



State isomorphism based symmetry reduction

■ Applications:

- Method is implemented in model-testing toolkit called NModel
(by Margus Veanes et al, MS Research Redmond)
- There is similar implementation developed independently by Arend Rensink in the GROOVE project



Iterated search refinement with bit-state pruning

■ Problem

- Calculate **reachability** in models that
 - Run out of resources without producing results with known methods
 - Produce too long traces to the error/desired state with known methods
- Such reachability problems can be used for solving problems related to
 - Scheduling
 - Hardware synthesis
 - Offline test generation

Iterated search refinement with bit-state pruning

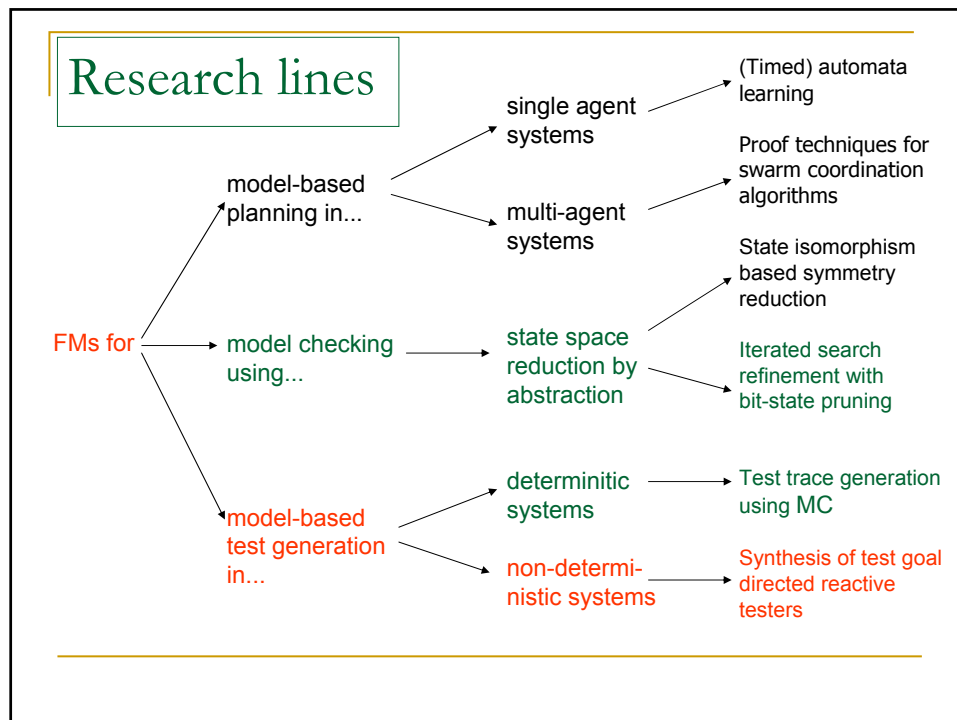
■ Solution

- Combine two well known methods in a new way:
 - Iterated Search Refinement, and
 - Bitstate hashing
- The key idea is to use collisions in bitstate hash table to randomly filter the search space.
 - Each iteration requires very small amounts of memory
 - A lot of prefixes of paths are covered multiple times, but
 - Provides good results in several examples.
- Results published in the PhD thesis of J. Ernits (Nov. 2007)

Iterated search refinement with bit-state pruning

■ Practical applications

- 2 well researched case studies:
 - Synthesis of a memory arbiter for a radar memory case study (J. Ernits, 2005)
 - Calculating offline test sequences for model-based testing (J. Ernits, A. Kull, K. Raiend, J. Vain, 2006)
 - The method cannot be used for disproving the reachability, but provides a new alternative for finding a witness trace.



Synthesis of test goal directed reactive testers

- **Problem:**
 - Given a
 - Non-deterministic EFSM M_{SUT} of System Under Test
 - testing goal in terms of a set \mathbf{Trp} of M_{SUT} transitions
 - Find EFSM M_{TSR} s.t.
 - M_{TSR} is I/O compliant with M_{SUT}
 - any trace of $M_{SUT} \parallel M_{TSR}$ includes labels of all transitions in \mathbf{Trp}
 - M_{TSR} chooses a transition from those labelled with current input from M_{SUT} that maximizes some gain function G .

Synthesis of test goal directed reactive testers

- Solution:
 - Algorithm of constructing M_{TSR}
- Complexity:
 - For all controllable transitions of the M_{TSR} the upper bound of the complexity of the computations of the gain functions is $O(|E_{SUT}|^3)$.
 - At runtime each choice by the tester takes $O(|E_{SUT}|^2)$ arithmetic operations to evaluate the gain functions
- (ASE2007, Vain, Raiend, Kull, Ernits)

Synthesis of test goal directed reactive testers

□ Experimental results

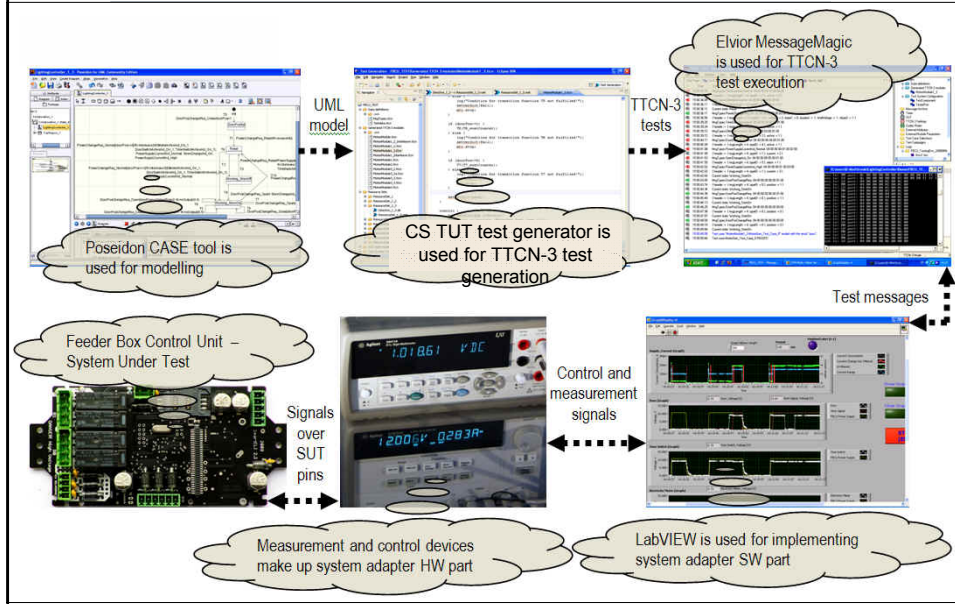
All Transition Test Purpose

Algorithm of the tester	Model 1 (8 trans.)	Model 2 (16 trans.)	Model 3 (32 trans.)
Random choice	56 ± 36	295 ± 130	1597 ± 1000
Anti-ant	21 ± 4	53 ± 13	218 ± 81
Reactive planner	17 ± 3	37 ± 6	80 ± 10

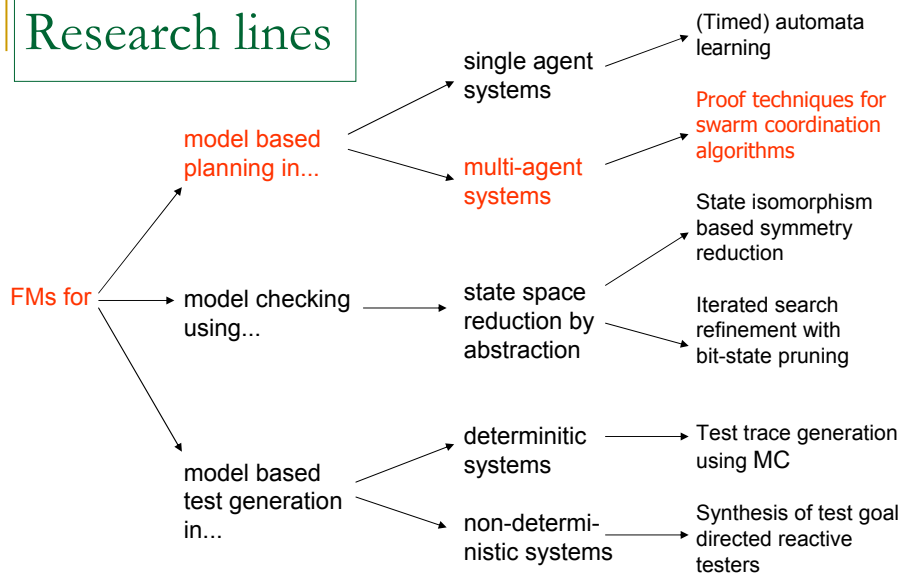
One Transition Test Purpose

Algorithm of the tester	Model 1 (8 trans.)	Model 2 (16 trans.)	Model 3 (32 trans.)
Random choice	34 ± 35	120 ± 114	699 ± 719
Anti-ant	14 ± 7	36 ± 19	140 ± 70
Reactive planner	5 ± 2	8 ± 3	11 ± 3

State-of-the-art



Research lines



Proof techniques for swarm coordination algorithms

- Problem:
 - proving properties of swarm (distributed) coordination algorithms
 - self-stabilization, coverage, ... can be reduced to reachability analysis
 - reachability analysis collides with scalability barrier

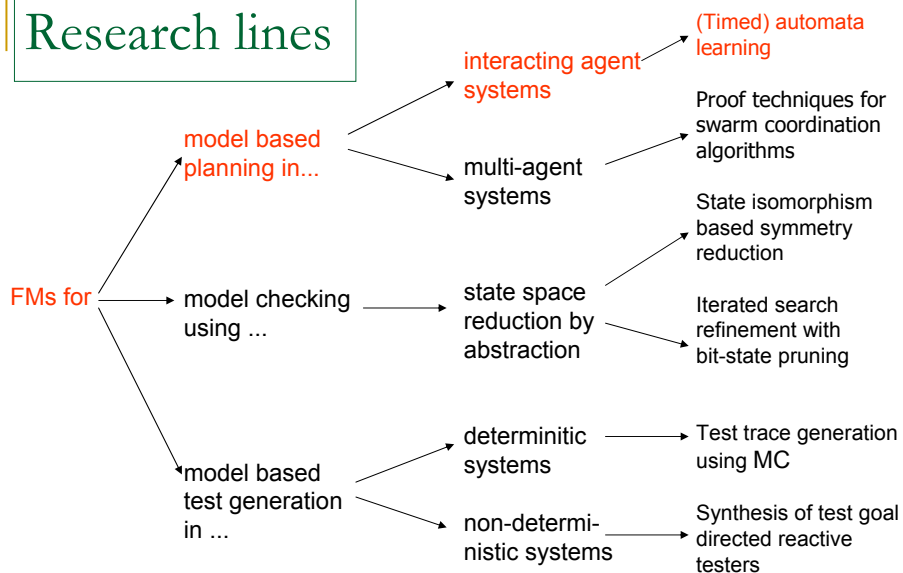
Proof techniques for swarm coordination algorithms (ongoing work)

- Some solutions:
 - Pattern based problem encoding provides right level of abstraction (spec. patterns are problem oriented)
 - General principles of proving:
 - Compositional approach combining different techniques of component proofs – MC, deduction,...
 - Global properties by structural induction on the size of swarm/task
 - Base: prove sufficiency of assumptions on a swarm fragment of some tractable size, e.g., using MC with symmetry reduction, (typically the size of fragment can't be trivial either)
(see BEC2008, Vain, Kuusik, Tammet, Juurik)
 - Step: Show that the induction step does not violate the assume part for the new fragment defined by the step

Proof techniques for swarm coordination algorithms

- Demo: *dynamic cleaning problem*
 - different zones of the room deteriorate with different rate
 - a swarm of cleaning robots should keep deterioration below treshold
 - robots of the swarm communicate trough *pheromone trace*
- Problem: given a treshold C_{Tresh} and initial value of the deterioration vector $\mathbf{S}^{det}(0)$, s.t. $S_i^{det}(0) > C_{Tresh}$ for all i ,
 - prove that
 - 1) the swarm is always able to reach the state $\mathbf{S}^{det}(t)$, s.t. $S_i^{det}(t) < C_{Tresh}$ for all i ,
and
 - 1) for all $t' > t$, condition $\mathbf{S}_t^{det}(t') < C_{Tresh}$ is invariantly true.

Research lines



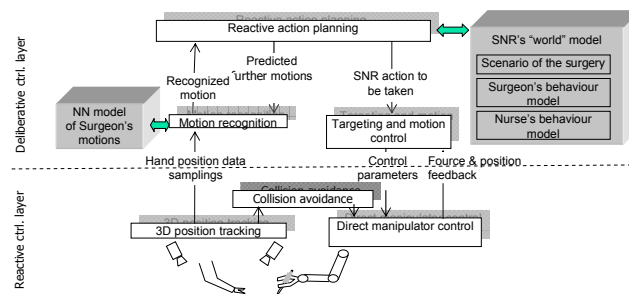
HRI automata learning

- Problem: Planner synthesis for a human adaptive Scrub Nurse Robot (SNR)
 - human adaptive = on-line learning from **H-H/H-R** interactions
 - learning = constructing/modifying a human motion model



(Timed) automata learning

- Context: SNR control architecture



(Timed) automata learning

Learning algorithm

Input:

- Time-stamped sequences of observations of HRI (set of timed traces $Tr^T(Obs)$)
- Parameters for
 - state rescaling operator - R
 - distinguishing model observables/controllables - $|_{TAIO}$
 - defining quotient state space

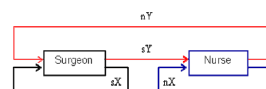
Output:

- Extended (Uppaal version) timed automaton TA s.t.
 $Tr^T(TA) = R(Tr^T(Obs)|_{TAIO}) / \sim$ % RTIOCO of timed traces

(Vain, Miyawaki, ICCAS-SICE2009, Fukuoka, Japan)

(Timed) automata learning (example)

Event	TS	Surgeon				Nurse				Notations:
		X _s	Y _s	Z _s	W _s	X _n	Y _n	Z _n	W _n	
Start	0	123	22	52	0	214	23	64	11	$X_s, X_n, Y_s, Y_n, Z_s, Z_n, W_s, W_n$ - coordinates of surgeon's and nurse's wrist
a_0^s	17					76	1	237	26	$X_s, Y_s, X_n, Y_n, Z_s, Z_n, W_s, W_n$ - normalized in interval [0,30], X_s, Y_s, X_n, Y_n coordinates
a_1^s	42					93	1	222	24	Z_s coordinates
a_2^s	48					57	1	191	21	Z_n coordinates
a_3^s	70	81	8	123	24	212	23	46	11	TS - switching event time stamp
a_4^s	78					132	1	245	27	Switching events of Nurse's Gestures:
a_5^s	79	118	20	85	6			26	23	a_0^n - idle
a_6^s	86	116	19	73	1			85	23	a_1^n - prepare instrument
a_7^s	88					73	1	202	22	a_2^n - picking up an instrument
a_8^s	107	121	21	59	1			44	2	a_3^n - holding the instrument & waiting
a_9^s	109					77	1	244	27	a_4^n - passing the instrument
a_{10}^s	122					86	1	259	29	a_5^n - wait returning
a_{11}^s	124	59	0	116	22	199	22	63	18	a_6^n - withdrawing hand
a_{12}^s	130	92	11	139	30	211	23	93	30	a_7^n - stretching hand
a_{13}^s	134					75	1	194	21	a_8^n - receiving
a_{14}^s	137					104	1	200	22	a_9^n - moving back
a_{15}^s	142	94	11	104	20	200	22	26	23	a_{10}^n - putting on the tray.
a_{16}^s	150	133	25	68	1	230	25	76	23	Switching events of Surgeon's gestures:
a_{17}^s	155	121	21	21	1			65	6	a_0^s - idle.
a_{18}^s	171	146	30	63	1			59	23	a_1^s - receiving an instrument.
a_{19}^s	177	138	27	108	18	170	18	52	23	a_2^s - inserting instrument.
a_{20}^s	180	147	30	66	1	169	18	62	23	a_3^s - working.
a_{21}^s	184					124	1	268	30	a_4^s - waiting for an instrument.
a_{22}^s	186					85	1	20	20	a_5^s - extracting from trocar cannula.
						50	1	20	20	a_6^s - returning the instrument
										a_7^s - working with i^{th} instrument. -- working with $i+1^{th}$ instrument





■ Thank You!