Task Scheduling and Execution for Long-Term Autonomy

Nick Hawes   Dave Parker

University of Birmingham

Part 2: Formal Guarantees for Robotic Navigation Planning

ICAPS Summer School, June 2016

Overview

- Formal verification
  - probabilistic model checking
- Markov decision processes (MDPs)
  - verification vs. strategy synthesis
- Linear temporal logic (LTL)
  - probabilistic model checking + MDPs + LTL
- Multi-objective probabilistic model checking
  - partially satisfiable task specifications

Formal verification

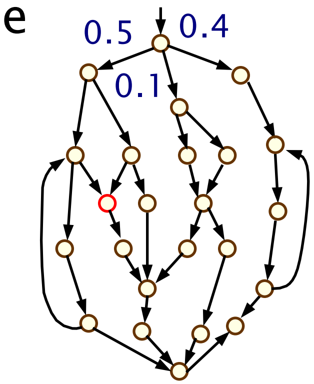
- Formal verification
  - the application of rigorous, mathematics-based techniques to check the correctness of computerised systems

- Verifying probabilistic systems…
  - unreliable or unpredictable behaviour
    - e.g. failures, message loss, delays, unreliable sensors/actuators
  - randomisation in algorithms/protocols
    - e.g. random back-off in communication protocols

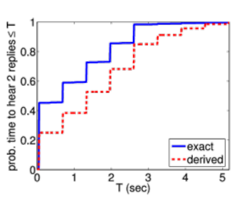
- We need to verify quantitative system properties
  - “the probability of the airbag failing to deploy within 0.02 seconds of being triggered is at most 0.001”
  - not just correctness: reliability, timeliness, performance, …
  - not just verification: correctness by construction

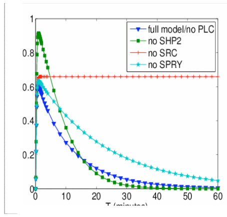
Probabilistic model checking

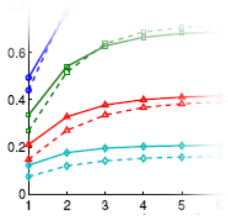
- Construction and analysis of probabilistic models
  - state-transition systems labelled with probabilities (e.g. Markov chains, Markov decision processes)
  - from a description in a high-level modelling language

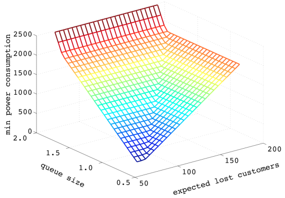


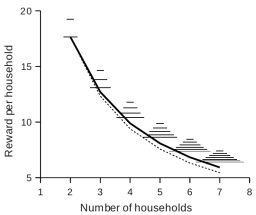
- Properties expressed in temporal logic, e.g. PCTL:
  - trigger $\rightarrow P_{\geq 0.999}$ [$F^{\leq 20}$ deploy]
  - "the probability of the airbag deploying within 20ms of being triggered is at at least 0.999"
  - properties checked against models using exhaustive search and numerical computation











Probabilistic model checking

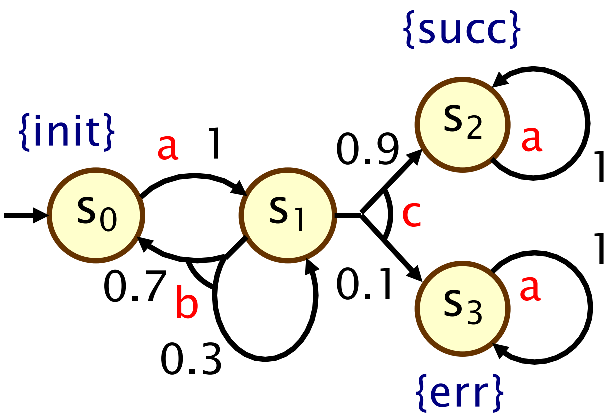
- Key benefits
  - exact results: guarantees, optimality, …
  - fully automated, tools available (e.g. PRISM)
  - wide range of models, properties expressible

- Key challenges
  - scalability! state space explosion problem
  - results are only as good as the model

- Application domains
  - network/communication protocols, security, biology, power management, robotics & planning, …

Markov decision processes (MDPs)

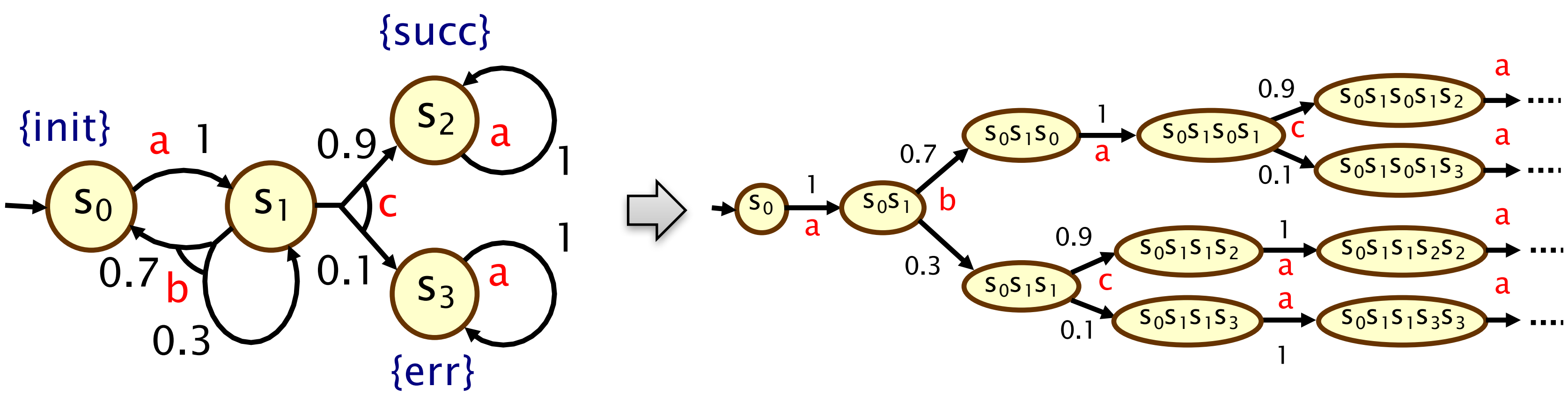
- Markov decision processes (MDPs)
  - model nondeterministic as well as probabilistic behaviour



- Nondeterminism for:
  - control: decisions made by a controller or scheduler
  - adversarial behaviour of the environment
  - concurrency/scheduling: interleavings of parallel components
  - abstraction, or under-specification, of unknown behaviour

Strategies

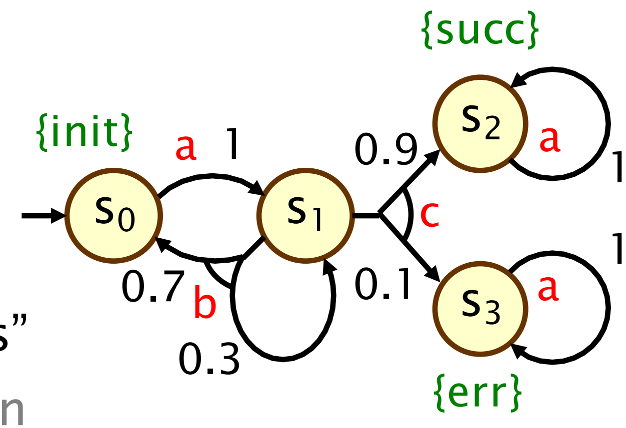
- A strategy (or "policy", "adversary", "scheduler")
  - is a resolution of nondeterminism, based on history
  - i.e. a mapping from finite paths to (distributions over) actions
  - induces (infinite-state) Markov chain (and probability space)



- Classes of strategies:
  - memory: memoryless, finite-memory, or infinite-memory
  - randomisation: deterministic or randomised

Verification vs. Controller synthesis

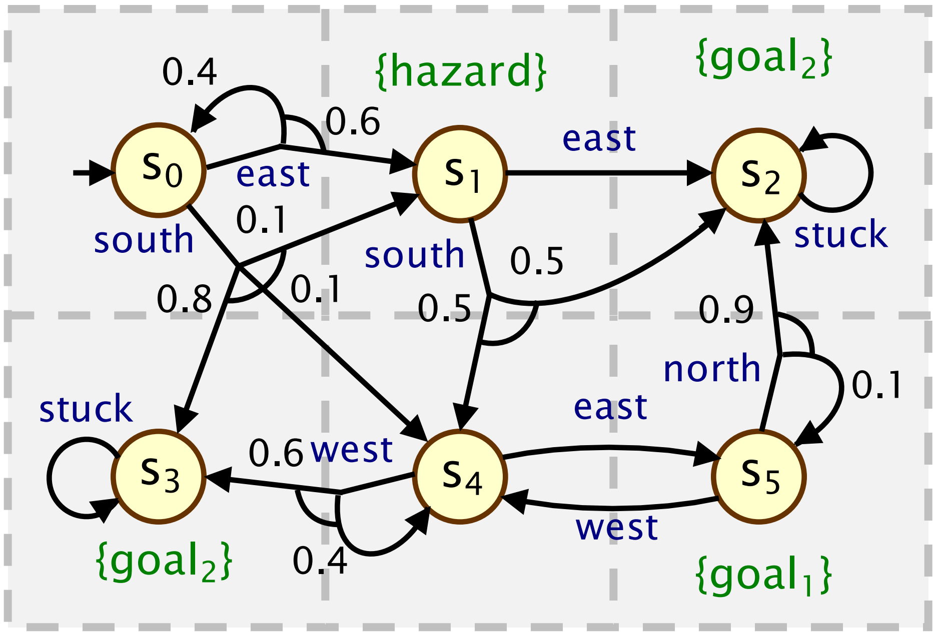
- 1. Verification
  - quantify over all possible strategies (i.e. best/worst-case)
  - $P_{\leq 0.1}$ [F *err*] : “the probability of an error occurring is $\leq$ 0.1 for all strategies”
  - applications: randomised communication protocols, randomised distributed algorithms, security, …



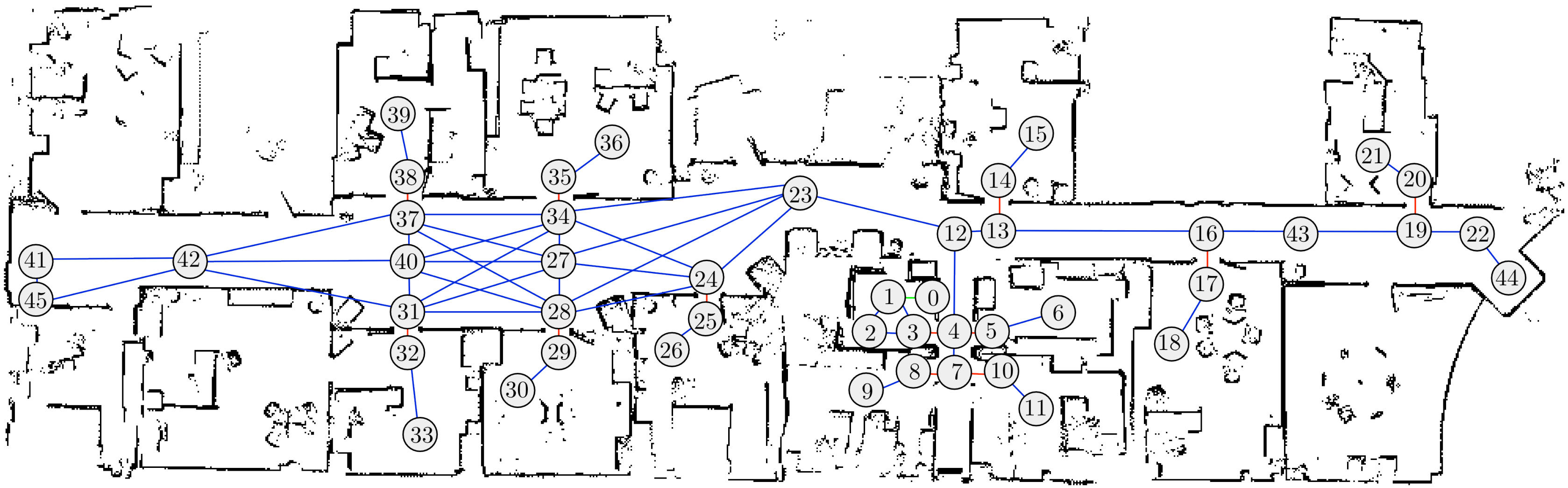
- 2. Controller synthesis
  - generation of "correct-by-construction" controllers
  - $P_{\leq 0.1}$ [F *err*] : "does there exist a strategy for which the probability of an error occurring is $\leq$ 0.1?”
  - applications: robotics, power management, security, …

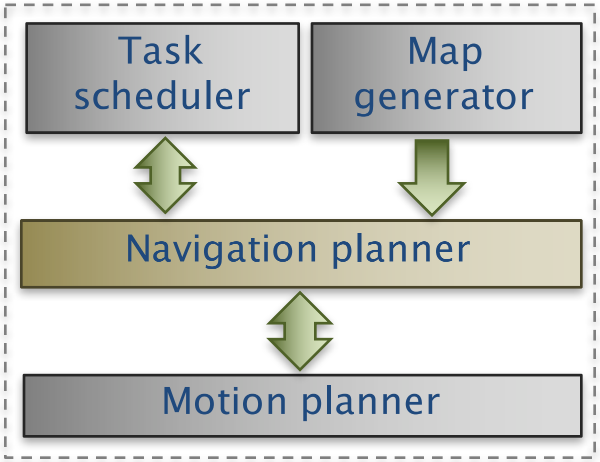
- Two dual problems; same underlying computation:
  - compute optimal (minimum or maximum) values

- Example MDP
  - robot moving through terrain divided in to 3 x 2 grid

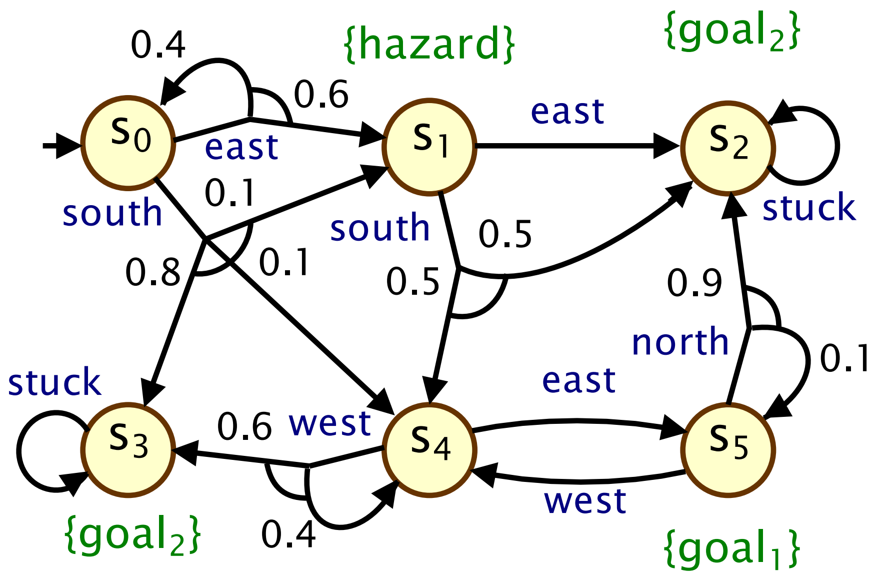


Larger example





Example – Reachability



Verify: $P_{\leq 0.6}$ [F goal$_1$]

or

Synthesise for: $P_{\geq 0.4}$ [F goal$_1$]

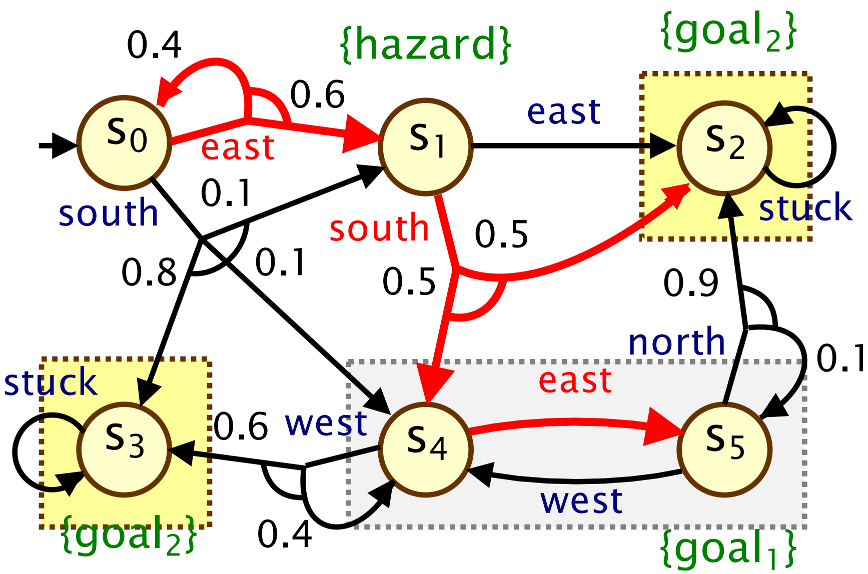
$\Downarrow$

Compute: $P_{max=?}$ [F goal$_1$]

Optimal strategies:
memoryless and deterministic

Computation:
graph analysis + numerical soln.
(linear programming, value iteration, policy iteration)

Example – Reachability



Optimal strategy:
s_0 : east
s_1 : south
s_2 : –
s_3 : –
s_4 : east
s_5 : –

Verify: $P_{\leq 0.6}$ [F goal$_1$]

or

Synthesise for: $P_{\geq 0.4}$ [F goal$_1$]

$\Downarrow$

Compute: $P_{max=?}$ [F goal$_1$] = 0.5

Optimal strategies:
memoryless and deterministic

Computation:
graph analysis + numerical soln.
(linear programming, value iteration, policy iteration)

Linear temporal logic (LTL)

- Logic for describing properties of executions [Pnueli]

- LTL syntax:
  - $\psi ::= \text{true} \mid a \mid \psi \wedge \psi \mid \neg\psi \mid X\,\psi \mid \psi\,U\,\psi \mid F\,\psi \mid G\,\psi$
- Propositional logic + temporal operators:
  - a is an atomic proposition (labelling a state)
  - $X\,\psi$ means "ψ is true in the next state"
  - $F\,\psi$ means "ψ is eventually true"
  - $G\,\psi$ means "ψ remains true forever"
  - $\psi_1\,U\,\psi_2$ means "ψ_2 is true eventually and ψ_1 is true until then"

- Simple examples
  - $G\neg(\text{critical}_1 \wedge \text{critical}_2)$ – "the two processes never enter the critical section simultaneously"
  - $\neg\text{error}\ U\ \text{end}$ – "the program terminates without any errors"

Linear temporal logic (LTL)

- LTL syntax:
  - $\psi ::= \text{true} \mid a \mid \psi \wedge \psi \mid \neg\psi \mid X\ \psi \mid \psi\ U\ \psi \mid F\ \psi \mid G\ \psi$

- Commonly used LTL formulae:
  - $G\ (a \rightarrow F\ b)$ – "b always eventually follows a"
  - $G\ (a \rightarrow X\ b)$ – "b always immediately follows a”
  - $G\ F\ a$ – "a is true infinitely often"
  - $F\ G\ a$ – "a becomes true and remains true forever"

- Robot task specifications in LTL
  - $(G \neg \text{hazard}) \wedge (G\ F\ \text{goal}_1)$ – "avoid hazard and visit goal_1 infinitely often"
  - $\neg \text{zone}_3\ U\ (\text{zone}_1 \wedge (F\ \text{zone}_4))$ – "patrol zone 1 then 4, without passing through 3".

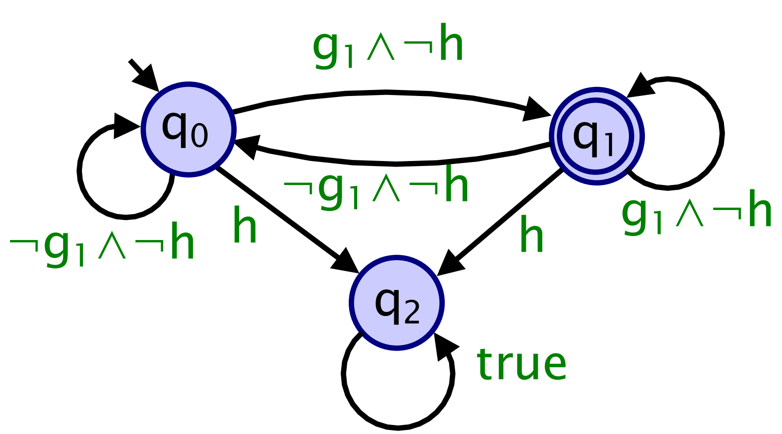
LTL for robot navigation

- Probabilistic LTL on MDPs
  - e.g. $P_{>0.7}$ [(G¬hazard) ∧ (GF goal$_1$)] – "is the probability of avoiding hazard and visiting goal$_1$ infinitely often > 0.7?"
  - e.g. $P_{max=?}$ [¬zone$_3$ U (zone$_1$ ∧ (F zone$_4$)] – "max. probability of patrolling zones 1 then 4, without passing through 3?"
- LTL + expected costs/times on MDPs
  - minimise expected time to satisfy (co-safe) LTL formulas
- Benefits of the approach
  - LTL: flexible, unambiguous property specification
  - guarantees on performance ("correct by construction")
  - meaningful properties: probabilities, time, energy,…
    - c.f. ad-hoc reward structures, e.g. with discounting
  - efficient, fully-automated techniques
    - LTL-to-automaton conversion, MDP solution

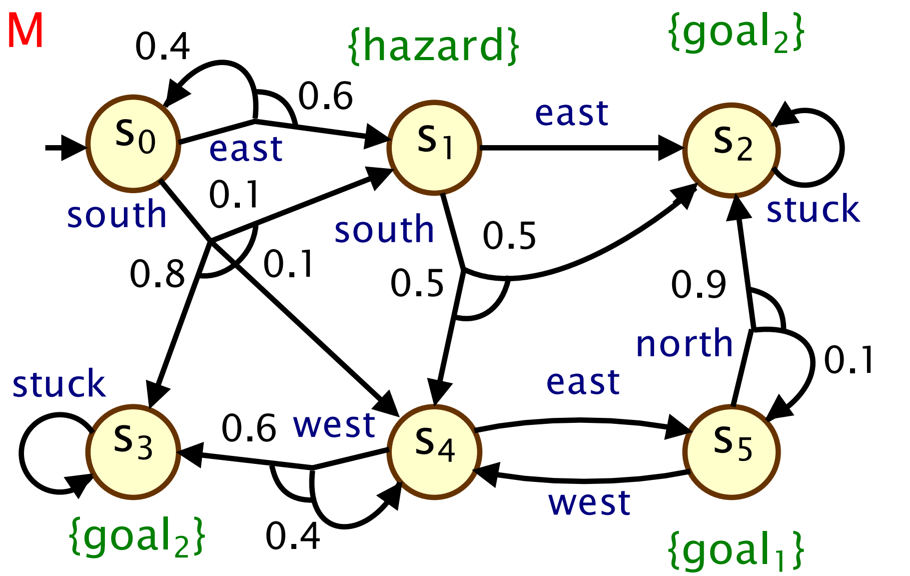
Probabilistic model checking LTL

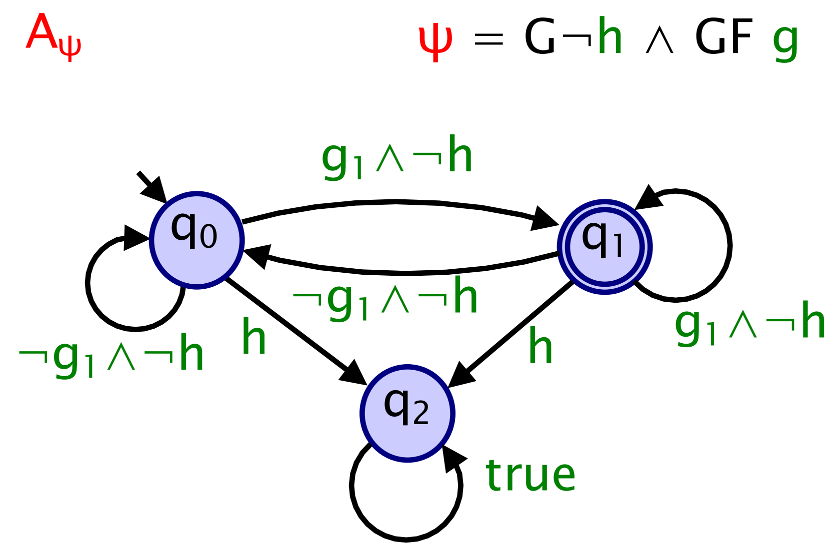
- Probabilistic model checking of LTL on MDPs
  - convert LTL formula ψ to deterministic automaton A_ψ (Buchi, Rabin, finite, …)
  - build/solve product MDP $M \otimes A_\psi$ (i.e. reduce to simpler problem)
  - optimal strategies are deterministic, finite-memory

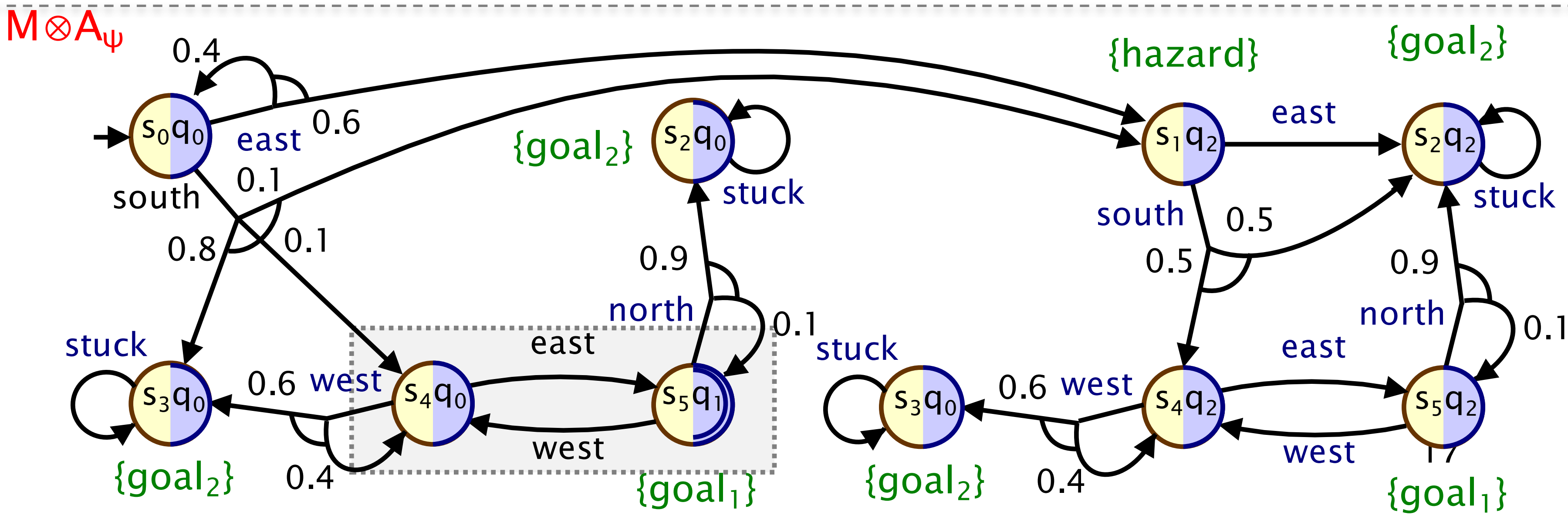
Det. Buchi automaton A_ψ
for $\psi = G\neg h \wedge GF\, g_1$



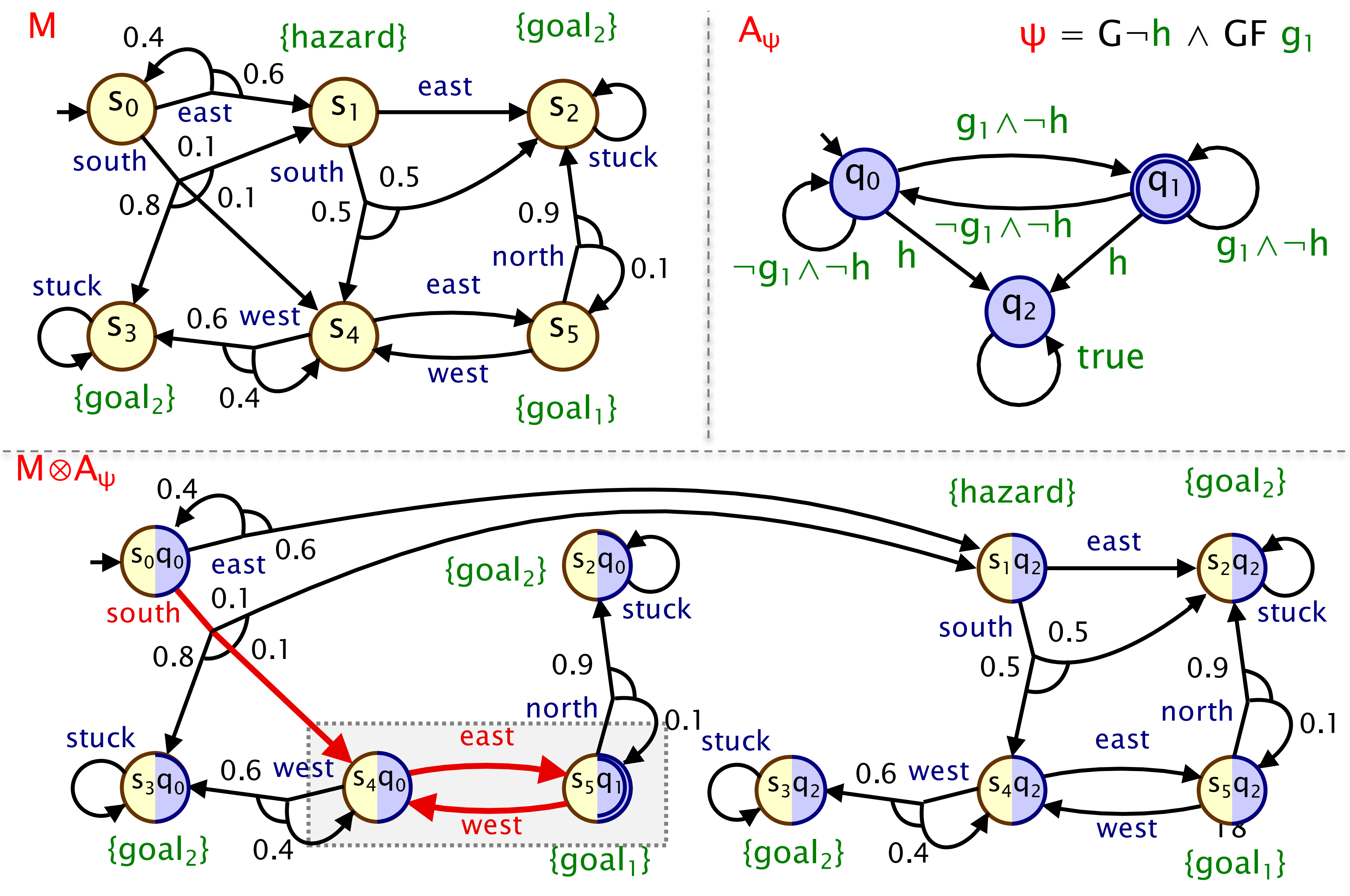
Example: Product MDP construction







Example: Product MDP construction

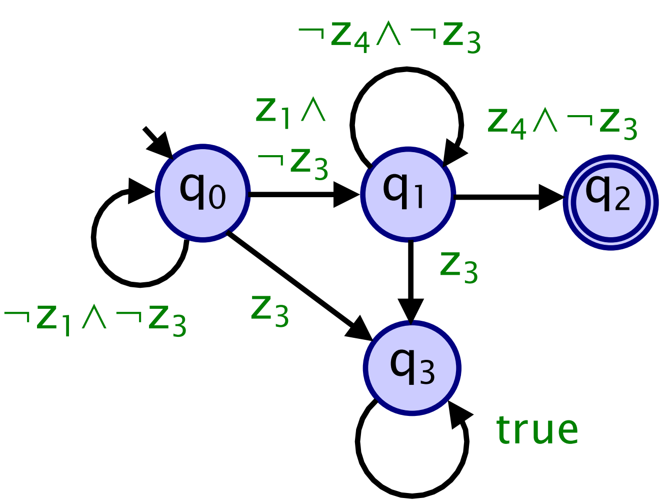


Co-safe LTL (and expected cost)

- Often focus on tasks completed in finite time
  - can restrict to co-safe fragment(s) of LTL
  - (any satisfying execution has a "good prefix")
  - e.g. $P_{max=?}$ [$\neg zone_3$ U ($zone_1 \wedge$ (F $zone_4$))]
  - for simplicity, can restrict to syntactically co-safe LTL

- Expected cost/reward to satisfy (co-safe) LTL formula
  - e.g. $R_{min=?}$ [$\neg zone_3$ U ($zone_1 \wedge$ (F $zone_4$))] – "minimise exp. time to patrol zones 1 then 4, without passing through 3".

- Solution:
  - product of MDP and DFA
  - expected cost to reach accepting states in product



Probabilistic model checking

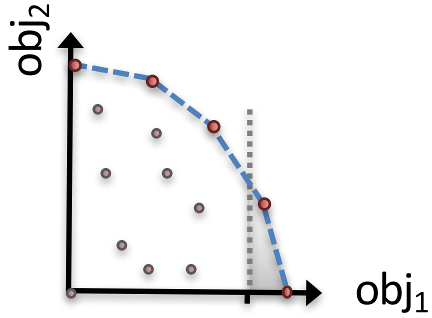
- Further use of probabilistic model checking…
  - (various probabilistic models, query languages)

- Nested queries
  - e.g. $R_{min=?}$ [safe U (zone$_1$ $\wedge$ (F zone$_4$)] – "minimise exp. time to patrol zones 1 then 4, passing only through safe".
  - where safe denotes states satisfying $\langle\langle$ctrl$\rangle\rangle$ $R_{<2}$ [F base] – "there is a strategy to return to base with expected time < 2"

- Analysis of generated controllers
  - expected power consumption to complete tasks?
  - conditional expectation, e.g. expected time to complete task, assuming it is completed successfully?
  - more detailed timing information (not just mean time)

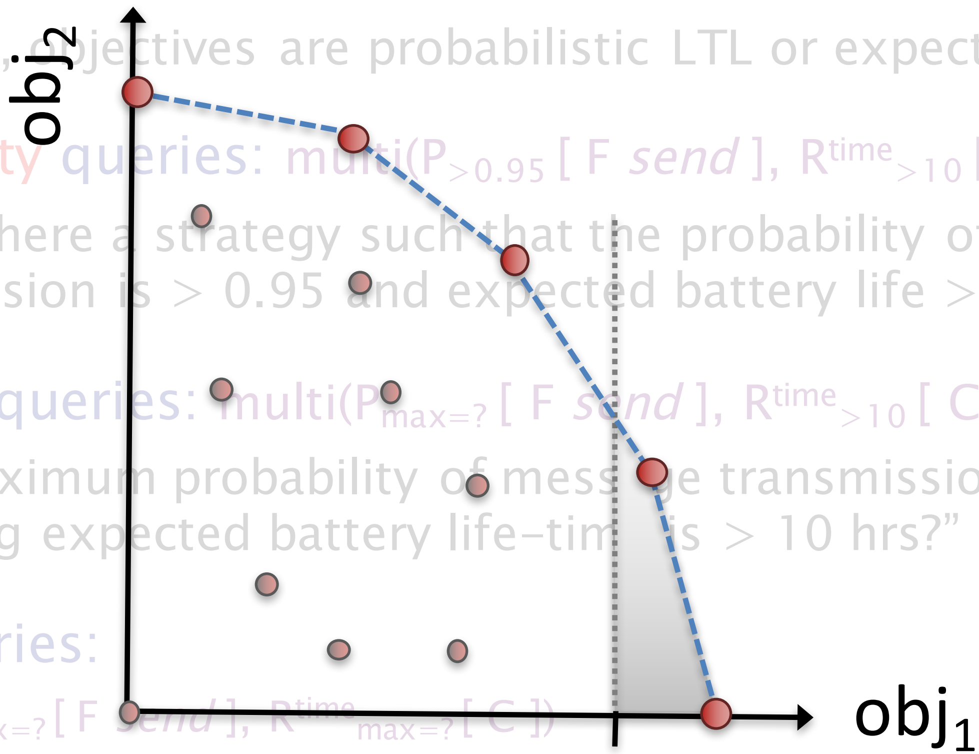
Multi-objective model checking

- Multi-objective probabilistic model checking
  - investigate trade-offs between conflicting objectives
  - in PRISM, objectives are probabilistic LTL or expected costs
- Achievability queries: multi($P_{>0.95}$ [F *send*], $R^{time}_{>10}$ [C])
  - e.g. "is there a strategy such that the probability of message transmission is > 0.95 and expected battery life > 10 hrs?"
- Numerical queries: multi($P_{max=?}$ [F *send*], $R^{time}_{>10}$ [C])
  - e.g. "maximum probability of message transmission, assuming expected battery life-time is > 10 hrs?"
- Pareto queries:
  - multi($P_{max=?}$ [F *send*], $R^{time}_{max=?}$ [C])
  - e.g. "Pareto curve for maximising probability of transmission and expected battery life-time"



Multi-objective model checking

- Multi-objective probabilistic model checking
  - investigate trade-offs between conflicting objectives
  - in PRISM, objectives are probabilistic LTL or expected rewards
- Achievability queries: multi($P_{>0.95}$ [F *send*], $R^{time}_{>10}$ [C])
  - e.g. “is there a strategy such that the probability of message transmission is > 0.95 and expected battery life > 10 hrs?”
- Numerical queries: multi($P_{max=?}$ [F *send*], $R^{time}_{>10}$ [C])
  - e.g. “maximum probability of message transmission, assuming expected battery life-time is > 10 hrs?”
- Pareto queries:
  - multi($P_{max=?}$ [F *send*], $R^{time}_{max=?}$ [C])
  - e.g. "Pareto curve for maximising probability of transmission and expected battery life-time”



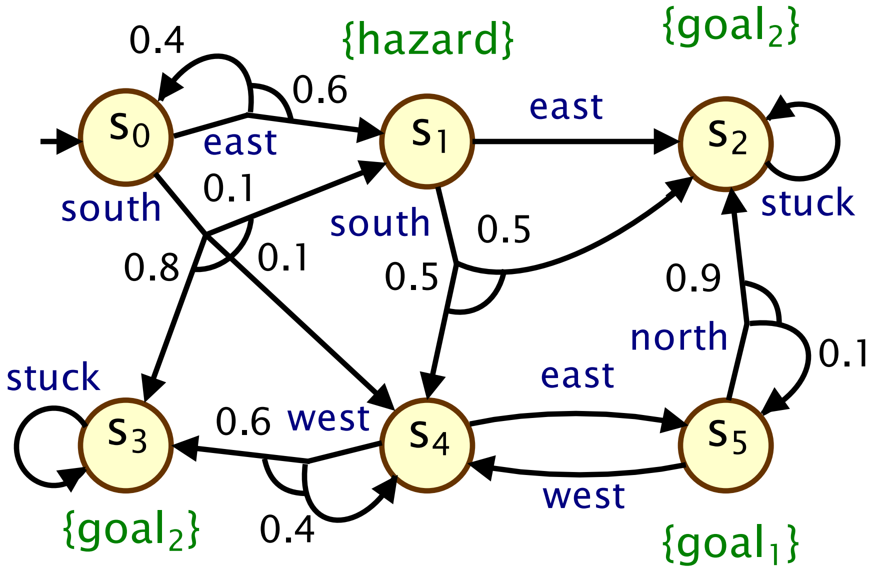
Multi-objective model checking

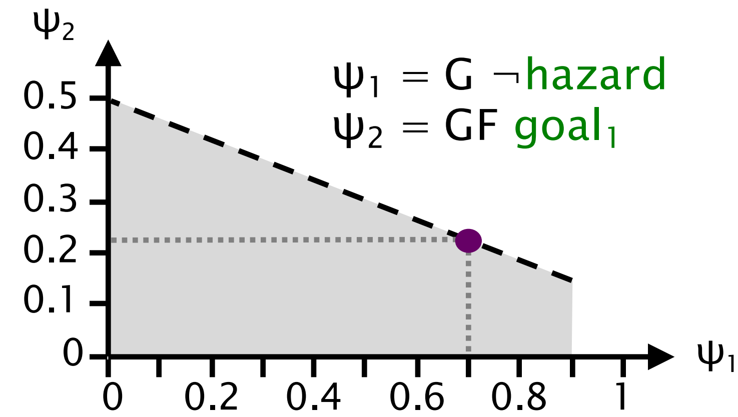
- Optimal strategies:
  - usually finite-memory (e.g. when using LTL formulae)
  - may also need to be randomised

- Computation:
  - construct a product MDP (with several automata), then reduces to linear programming [TACAS'07,TACAS'11]
  - can be approximated using iterative numerical methods, via approximation of the Pareto curve [ATVA'12]

- Extensions [ATVA'12]
  - arbitrary Boolean combinations of objectives
    - e.g. $\psi_1 \Rightarrow \psi_2$ (all strategies satisfying ψ_1 also satisfy ψ_2)
    - (e.g. for assume-guarantee reasoning)
  - time-bounded (finite-horizon) properties

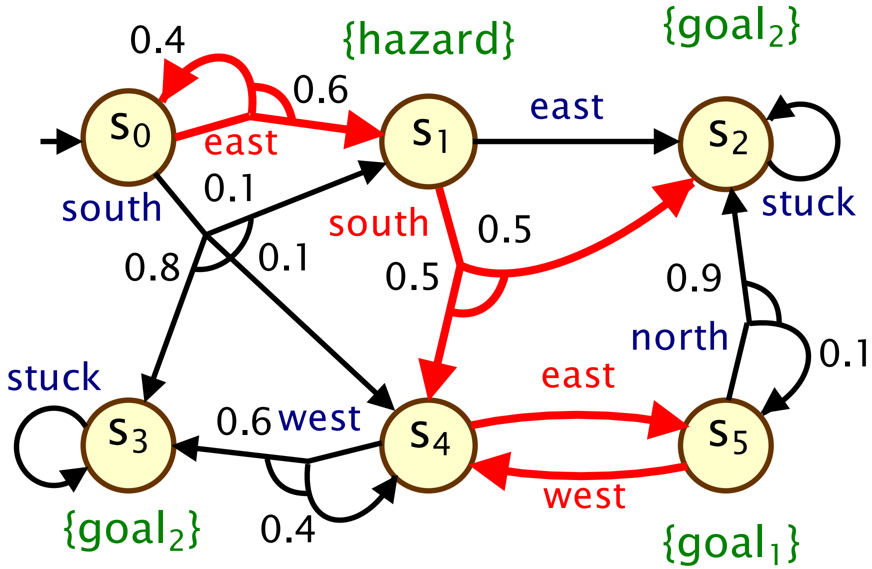
Example – Multi-objective





- Achievability query
  - $P_{\geq 0.7}$ [G ¬hazard] ∧ $P_{\geq 0.2}$ [GF goal$_1$] ? True (achievable)
- Numerical query
  - $P_{max=?}$ [GF goal$_1$] such that $P_{\geq 0.7}$ [G ¬hazard] ? ~0.2278
- Pareto query
  - for $P_{max=?}$ [G ¬hazard] ∧ $P_{max=?}$ [GF goal$_1$] ?

Example – Multi-objective



Strategy 1
(deterministic)

s_0 : east

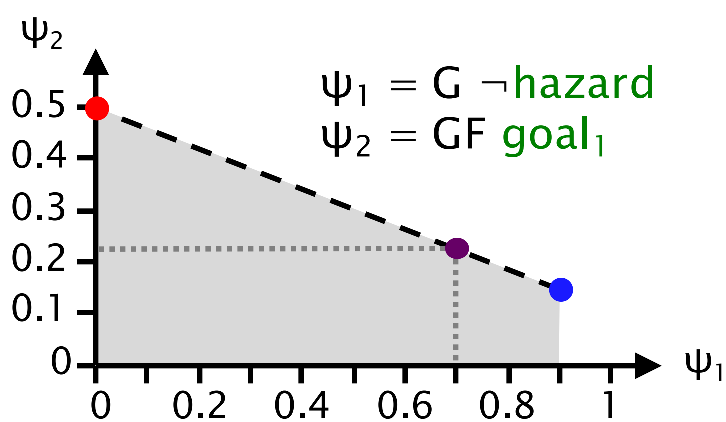
s_1 : south

s_2 : –

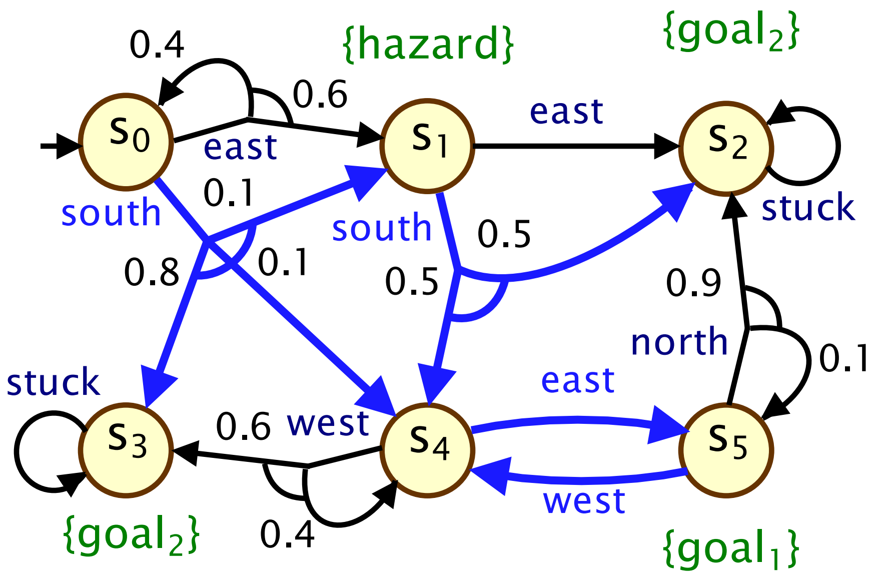
s_3 : –

s_4 : east

s_5 : west



Example – Multi-objective



Strategy 2
(deterministic)

s_0 : south

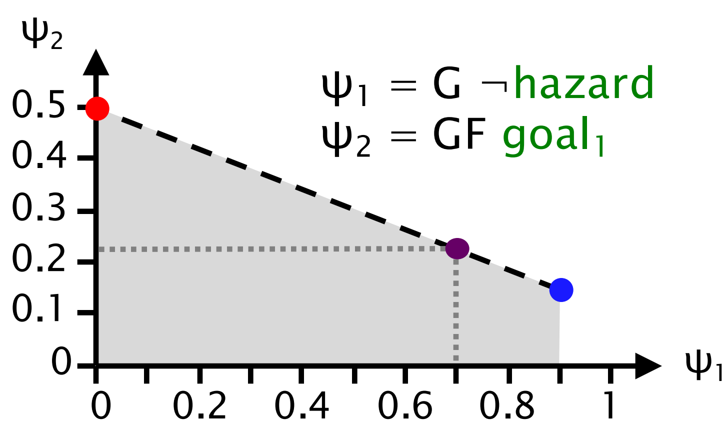
s_1 : south

s_2 : –

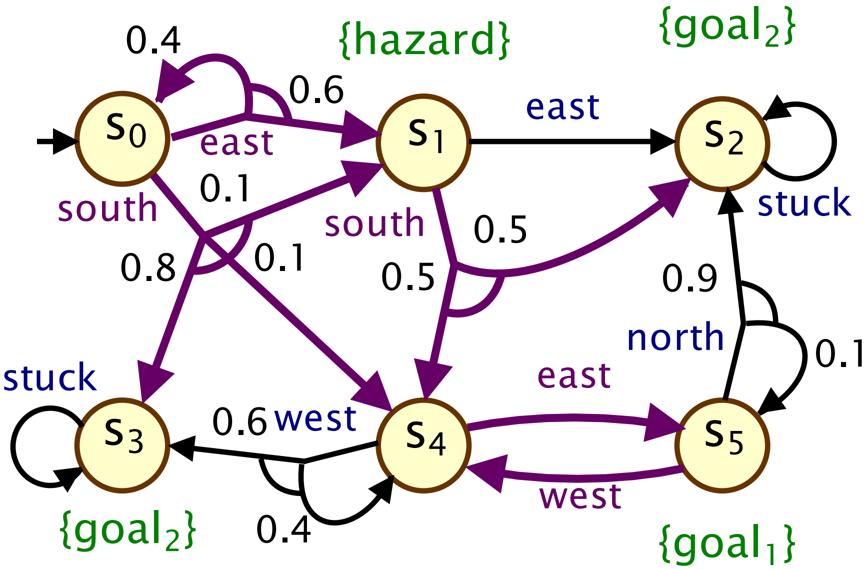
s_3 : –

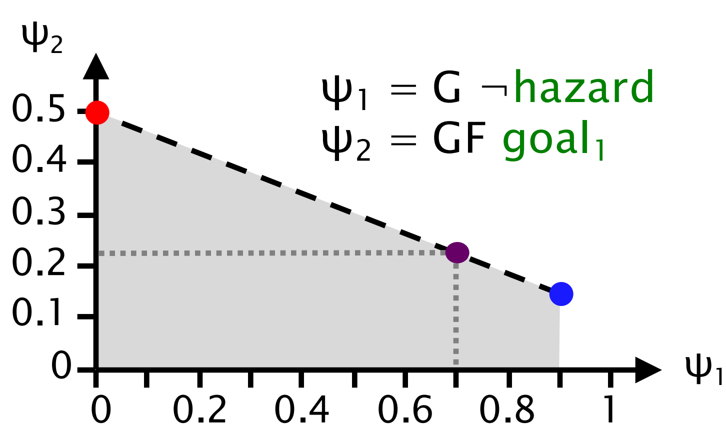
s_4 : east

s_5 : west



Example – Multi-objective





Optimal strategy:
(randomised)

s_0 : 0.3226 : east
0.6774 : south
s_1 : 1.0 : south
s_2 : –
s_3 : –
s_4 : 1.0 : east
s_5 : 1.0 : west

Application: Partially satisfiable tasks

- Partially satisfiable task specifications
  - via multi-objective probabilistic model checking [IJCAI'15]
  - e.g. $P_{max=?}$ [$\neg zone_3$ U ($room_1$ $\wedge$ (F $room_4$ $\wedge$ F $room_5$)] < 1
- Synthesise strategies that, in decreasing order of priority:
  - maximise the probability of finishing the task;
  - maximise progress towards completion, if this is not possible;
  - minimise the expected time (or cost) required
- Progress metric constructed from DFA
  - (distance to accepting states, reward for decreasing distance)
- Encode prioritisation using multi-objective queries:
  - $p = P_{max=?}$ [task]
  - r = multi($R^{prog}_{max=?}$ [C], $P_{>=p}$ [task])
  - multi($R^{time}_{min=?}$ [task], $P_{>=p}$ [task] $\wedge$ $R^{prog}_{>=r}$ [C])

Conclusion

- Rigorous probabilistic guarantees for robot navigation
  - formal verification + probabilistic model checking
  - Markov decision processes (MDPs)
  - linear temporal logic (LTL)
  - multi-objective model checking

- More details
  - Lacerda/Parker/Hawes. Optimal & Dynamic Planning for Markov Decision Processes with Co-Safe LTL Specifications, IROS'14
  - Lacerda/Parker/Hawes. Optimal Policy Generation for Partially Satisfiable Co-Safe LTL Specifications, IJCAI'15
  - PRISM: www.prismmodelchecker.org