COMS 412: Final Project

Various Concurrency Experiments with Probabilistic Model Checking

Charles Dudley

Introduction

For safety critical systems, testing is simply an insufficient method of assessing correctness. Testing is a very limited method of analysis in that it can only find the bugs you look for. A robust test suite can be generated for a system which still misses some critical error in implementation. For *safety critical systems* (systems where failure is extremely costly), we cannot afford for errors to slip past our testing. Examples might include an air traffic control system, where any fault could cost hundreds lives, or the software controlling a satellite launched away into space, which could cost untold millions of dollars if it were to crash.

Formal Verification is a powerful tool used in verifying the correctness and reliability of safety critical systems which can account for the shortcomings of traditional testing. Formal Verification techniques such as *Model Checking* are a more robust method of analysis that involves the exploration of a system's entire state space. These techniques can be used to discover errors in implementation and/or specification which may easily be missed by any test generation technique. When applied effectively, model checking can provide absolute proofs of an implementation's alignment to given specifications (these determinations are referred to as *correctness*). A model checker's proof of correctness can be a very powerful safety guarantee for smaller systems, but for many systems it is very difficult (or not practical) to achieve such a guarantee. Firstly, for many systems it is simply impossible to explore the entire state space with a reasonable amount of compute and memory. Models of systems can easily contain trillions of states, in which case verification of a property over the entire system is very difficult. These systems still may be analyzed by applying simplifications to the model of the system, but the guarantees resulting from verification on the simplified model may not transfer to the real system. Additionally, many systems involve a level of uncertainty that clouds the definitive *yes/no* property satisfaction we would like to assert on systems. The behavior of a system is likely influenced by the environment it operates in. For instance, the performance of a self-driving car is affected by the accuracy of its computer vision processing (a probabilistic process itself), the conditions of the road surface, and behavior of other (potentially manually operated) vehicles. These external influences on the behavior of a system are very difficult to capture in traditional verification methods, but may be extremely valuable in assessing the safety of that system. These systems may still be analyzed by applying certain assumptions to the environment, such as modeling the road surface conditions as a markovian process.

In this report I present a study of multiple applications of probabilistic model checking to real world scenarios, such as air traffic control systems. I use PRISM model checker[**?**] and PRISM specification language to model and analyze these systems. For each system I provide quantitative analysis of safety properties.

Terminology

1. **Discrete-Time Markov Chain**:

A *Discrete-Time Markov Chain* (DTMC) is a stochastic process which models the behavior of some process external to our *agent*. DTMCs are represented as finite state machines where every transition is assigned a probability value between 0 and 1. For a DTMC, a transition from one state to another is controlled entirely by the transition probabilities of the system. Behavior of this model can be predicted and analyzed without any modification.

DTMC M_c is formally defined as $M_c : (S, \sigma, s_0, \Sigma, L)$ with

- Probability transition function, $\sigma : S \times S \to [0, 1]$
- Initial state, s_0

- Set of atomic propositions, Σ
- Labeling function, L

2. 1. **Agent**: A decision-making entity in a scenario. The agent is capable of making non-deterministic decisions, called *actions*.

3. **Markov Decision Process**:
A *Markov Decision Process* (MDP) is a stochastic process modeling the behavior of a *non-deterministic agent*. MDPs are represented as finite state machines with each transition assigned an *action* as well as a probability value between 0 and 1.
An agent (modeled with an MDP) must *non-deterministically* select an *action* with every state transition. This action may affect the probability transition function, therefore behavior cannot be predicted in isolation. Markov Decision Process M_p is formally defined as $M_p : (S, s_0, A, \sigma, \Sigma, L)$ with

- Set of states, S
- Initial state, $s_0 \in S$
- Finite set of actions, A
- Probability transition function, $\sigma : S \times A \times S \to [0, 1]$
- Set of atomic propositions, Σ
- Labeling function, L

4. **Policy**: A rule, which when applied to an MDP, resolves the non-determinism in the system. A policy is essentially a list of rules which the agent must follow, specifying an action the agent must take under certain conditions. Application of a policy to an MDP reduces it to a deterministic system which we can reason about more effectively.

5. **Parallel Composition**: A system which models the joint behavior of multiple (synchronized) independent systems.

A parallel composition of an MDP, $M_p : (S_p, s_{0p}, A, \sigma_p, \Sigma_p, L_p)$ and an DTMC, $M_c : (S_c, \sigma_c, s_{0c}, \Sigma_c, L_c)$, is defined as $M_{||} : (S, s_{00}, A, \sigma, \Sigma, L)$ with

- Set of states, $S = S_p \times S_c$
  - State $(s_p, s_c) \in S$ may be abbreviated to s_{pc}
- Initial state, $s_{00} = (s_{0p}, s_{0c}) \in S$
- Finite set of actions, A
- Probability transition function, $\sigma : S \times A \times S \to [0, 1]$
  - $\sigma(s_{ix}, a, s_{jy}) = \sigma_p(s_i, a, s_j) \times \sigma_c(s_x, s_y)$
- Set of atomic propositions, $\Sigma = \Sigma_p \cup \Sigma_c$
- Labeling function, $L : S \to \Sigma$

A parallel composition of an MDP, $M_x : (S_x, s_x, A_x, \sigma_x, \Sigma_x, L_x)$ and another MDP, $M_y : (S_y, s_y, A_y, \sigma_y, \Sigma_y, L_y)$, is defined as $M_{||} : (S, s_{xy}, A, \sigma, \Sigma, L)$ with

- Set of states, $S = S_x \times S_y$
  - State $(s_x, s_y) \in S$ may be abbreviated to s_{pc}
- Initial state, $s_{xy} = (s_x, s_y) \in S$
- Finite set of actions, $A = A_x \cup A_y$
- Probability transition function, $\sigma : S \times A_x \times A_y \times S \to [0, 1]$
  - $\sigma(s_{im}, a_x, a_y, s_{jn}) = \sigma_x(s_i, a_x, s_j) \times \sigma_y(s_m, a_y, s_n)$
- Set of atomic propositions, $\Sigma = \Sigma_p \cup \Sigma_c$
- Labeling function, $L : S \to \Sigma$

Tools

PRISM

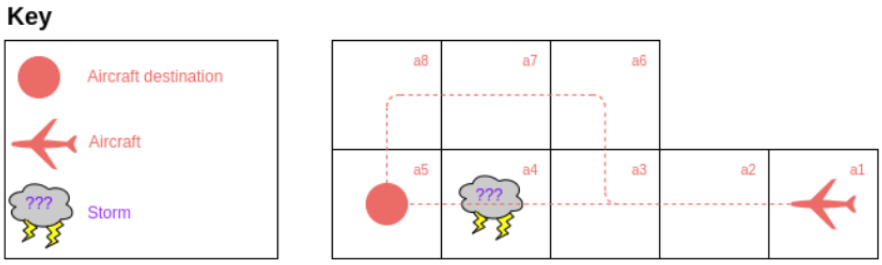
PRISM is both a *specification language* and a *probabilistic model checker*. Both forms of PRISM are used in this project to conduct probabilistic analysis on a series of probabilistic systems.
TODO!!! Explain auto parallel comp and specs

Overview of Scenarios

Scenario 1: Around the Storm

For this scenario we will consider a single non-deterministic agent and its interaction with a probabilistic environment.

Visualization:



Description:
An *aircraft* is travelling along a planned trajectory at a constant rate. The planned trajectory, T, follows along locations $[a_1, a_2, a_3, a_4, a_5]$ (without making any maneuvers), which is the optimal path from initial location, a_1, to destination, a_5.

There is a certain probability that a *storm* will occur along the planned trajectory (in location a_3), making the planned trajectory dangerous for air travel. In this case, the aircraft should maneuver to follow a safe trajectory, T'. This alternate trajectory traverses along locations $[a_1, a_2, a_3, a_6, a_7, a_8, a_5]$.

In order to minimize cost, trajectory T is preferred. However, a maneuver should be executed by the aircraft if it is on course for dangerous airspace (the storm).

Aside: Although very simple, this model can provide interesting insights into expected behavior of a probabilistic system. Results and practices from this example may be expanded upon to model much more interesting scenarios, as will be explored in later sections.

Specifications:
In order to conduct analysis over this system, specifications of desired/undesired behavior must be drafted. A few examples of English specifications:

- The aircraft always eventually reaches its destination.
- The aircraft reaches its destination without entering the storm.
- The aircraft reaches its destination without entering the storm while only changing course to avoid a storm.
- The aircraft enters the storm.

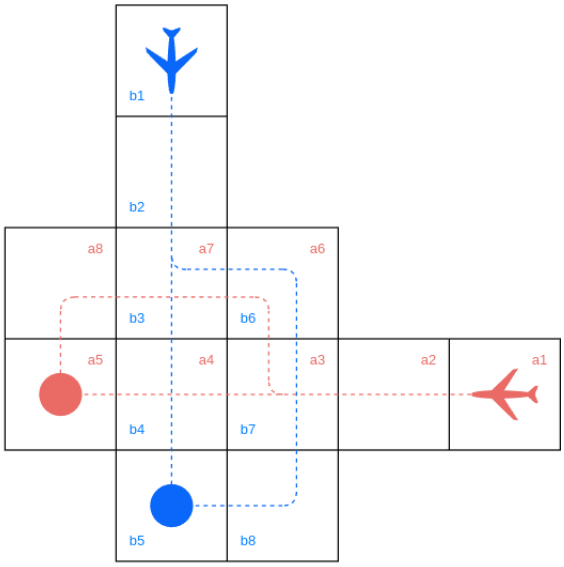
For probabilistic analysis, these specifications must be revised to the following form:

- What is the maximum probability the aircraft always eventually reaches its destination?
- What is the maximum probability the aircraft can reach its destination without entering the storm?
- What is the maximum probability the aircraft can reach its destination without entering the storm, given it never changes course except to avoid a storm?
- What is the maximum probability the aircraft enters the storm?

Scenario 2: Air Traffic Control

For this scenario we will consider the interaction between two non-deterministic agents.

Visualization:



Description:
An aircraft, *pink*, is travelling along a planned trajectory at a constant rate. The planned trajectory, T_p, follows along locations $[a_1, a_2, a_3, a_4, a_5]$ (without making any maneuvers), which is the optimal path from initial location, a_1, to destination, a_5.

Another similar aircraft, *blue*, is also travelling along its planned trajectory at the same rate. This planned trajectory, T_b, follows along locations $[b_1, b_2, b_3, b_4, b_5]$ (without making any maneuvers), which is the optimal path from initial location, b_1, to destination, b_5.

Trajectories T_p and T_b intersect in a_4/b_4, creating a potential collision. This collision is highly likely due to the aircraft's initial speeds and distances from a_4/b_4. In order to avoid a potential collision, either aircraft may maneuver into their alternate trajectory. For *pink* this alternate trajectory, T'_p, is $[a_1, a_2, a_3, a_6, a_7, a_8, a_5]$. For *blue* this alternate trajectory, T'_b, is $[b_1, b_2, b_3, b_6, b_7, b_8, b_5]$.

Similarly to the Storm Avoidance scenario, each aircraft should prefer their initial trajectory in navigating to their destination to minimize cost. However, a maneuver must be executed if a collision is imminent.

Specifications:
Some interesting specifications for analysis of this system:

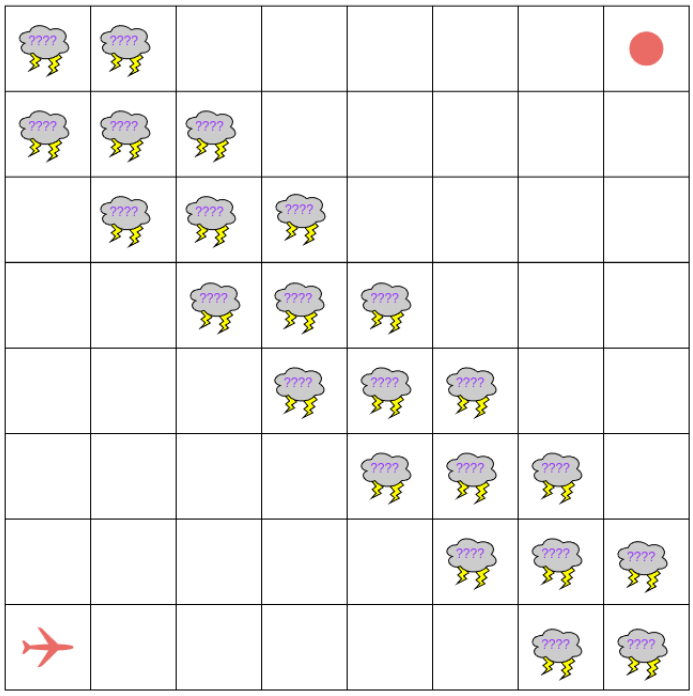
- Each aircraft reaches its respective destination without a collision.
- Each aircraft reaches its respective destination without a collision, executing a maneuver only to avoid an imminent collision.
- A collision eventually occurs in cell $a_x b_y$.
- A collision occurs.

Similarly to the Storm Avoidance scenario, these specifications must be rewritten into an inquisitive form (*What is the maximum probability of X?*) to allow for probabilistic analysis of these properties.

Scenario 3: Through The Storm

For this scenario we will, again, consider the interaction between a single non-deterministic agent and its deterministic environment. However, in this case the agent and environment are both much more complicated than in previous scenarios.

Visualization:



Description:
An *aircraft* is initially travelling East from position (0,0). In order to reach its destination, the aircraft must navigate from its initial position to its destination, (8,8). This aircraft is capable of turning to travel Northward or Eastward, or continuing in its current direction, but it is forbidden from turning in any other direction.

In order to reach its destination, the aircraft must pass through a *storm* which spans the entire airspace. Due to the severity of the storm, the aircraft may only sustain itself for a limited period of time inside of the storm. Therefore it is critical that the aircraft navigate through the dangerous airspace while minimizing its time spent inside the storm.

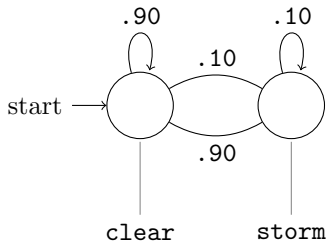
Specifications:
Some interesting probabilistic specifications for analysis of this system:

- What is the maximum probability of reaching the destination?
- What is the maximum probability of reaching the destination without entering a storm?
- What is the maximum probability of reaching the destination while minimizing contact with storms below some threshold, ϕ?

Modeling the Scenarios

Scenario 1: Around The Storm

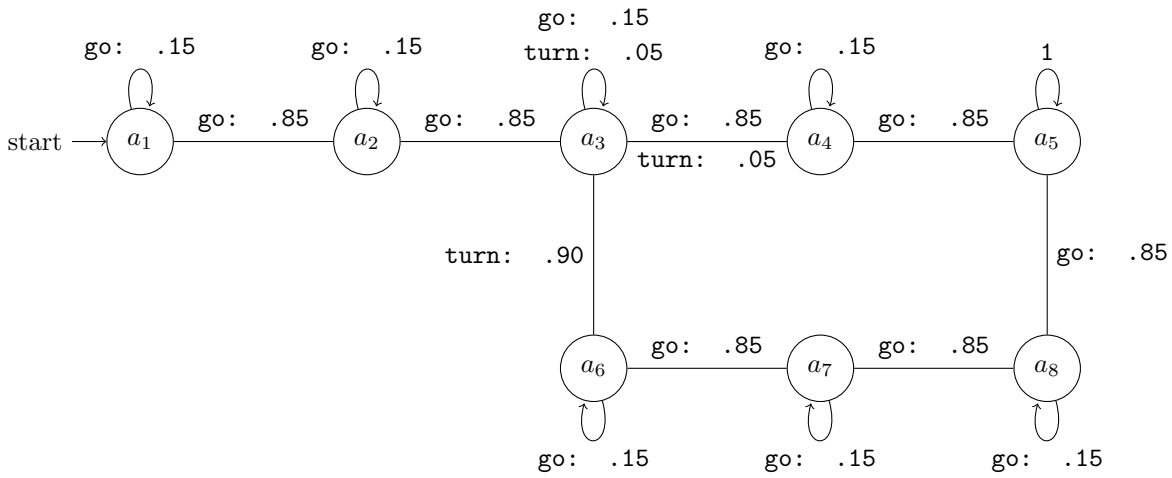
The behavior of the ***storm*** will be modeled by a DTMC, M_c as follows:



This DTMC has two states, corresponding to the status of the airspace in a_4: *clear* or *stormy* skies. The skies are initially *clear* and tend to change state with low probability (0.10).

Aside: This is clearly a gross oversimplification of the behavior of a real storm, however simplifications of this magnitude must be applied in order to create a reasonably sized model a system (in fact, even an entity as simple as this one can create state space explosion, as will be detailed in a later section). Simplifications of this sort will be applied to entities in all scenarios, including both *storms* and *aircraft*.

The behavior of the ***aircraft*** will be modeled by an MDP, M_p as follows:



This MDP has 8 states, representing each of the a_i locations the aircraft may travel through.

The two *non-deterministic actions* available to this agent are `go` and `turn`.

`go` is an action whose execution results in high probability (0.85) that the aircraft will move directly from its current state a_i, to the next state a_{i+1}, and a low probability (0.15) that the aircraft will remain in a_i. `go` is enabled in all states except the destination, a_5.

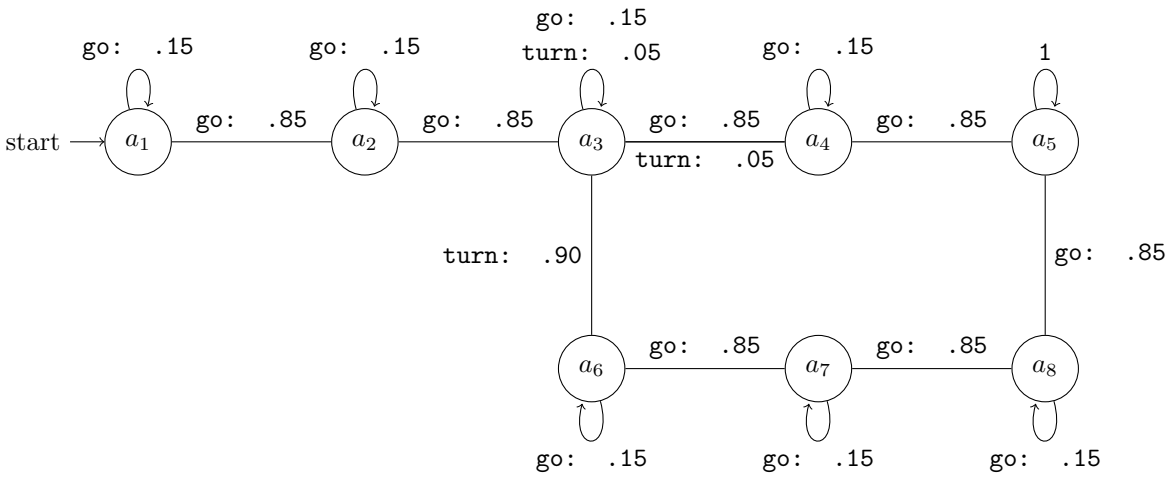
`turn` is an action whose execution results in high probability (0.90) of the aircraft altering its trajectory, and a low probability of either remaining in its current position or moving ahead (0.50 for either case). This action is defined only in a_3.

The ***joint behavior*** of the two independent systems, M_c and M_p, can be modeled using a *parallel composition* of the two systems. This parallel composition will be defined as an MDP, M, constructed according to the definition given above in **Terminology**.

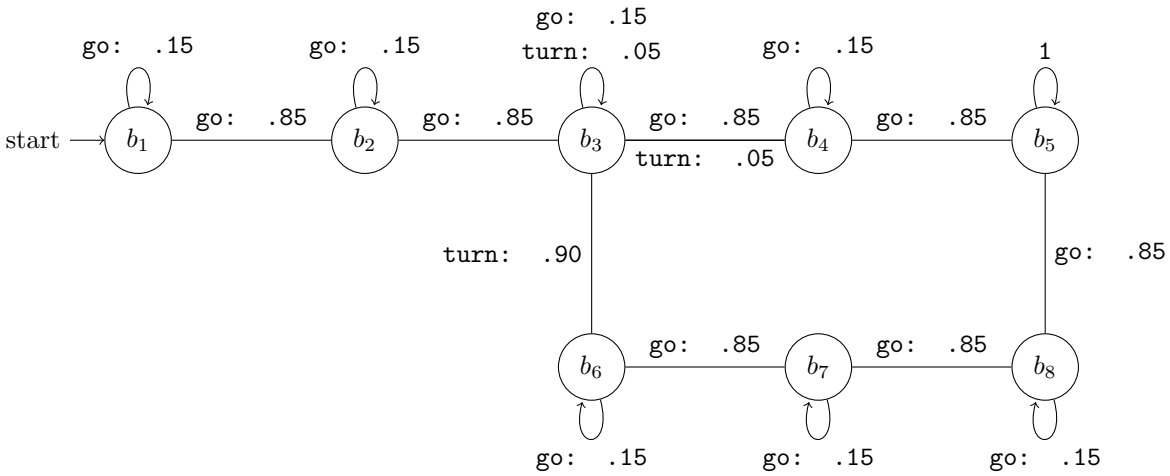
For the sake of brevity, a depiction of this parallel composition is not provided.

Scenario 2: Air Traffic Control

The behavior of ***pink*** will be modeled by an MDP as follows:



The behavior of ***blue*** will be modeled by an identical MDP as follows:



The *non-deterministic actions* available to ***pink*** and ***blue*** are identical to those of the ***aircraft*** from **Around The Storm** above.

The ***joint behavior*** of these two independent systems can also be modeled using a *parallel composition*. This resulting system will contain 64 states (8 states in *pink* $\times$ 8 states in *blue*). Additionally, the actions available to this composed system will be pairs of actions selected in each subsystem, ex. `pink_go_blue_go`, `pink_turn_blue_go`, etc. (we might think of this composed system as coordinating decision making between the two independent agents). Details of the parallel composition are defined in **Terminology**.

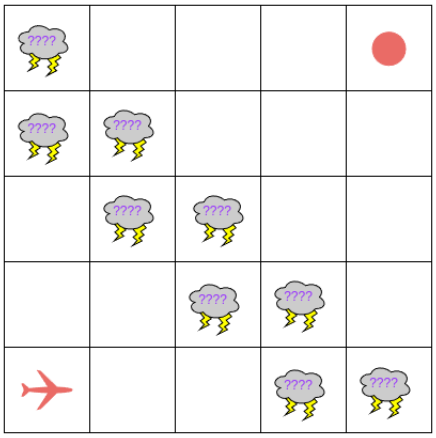
Again, for the sake of brevity the parallel composition is not depicted for this system.

Scenario 3: Through The Storm

The behavior of the ***storm*** will be modeled by a parallel composition of many independent ***storm cells***. Each cell is modeled by a DTMC, M_i, where each M_i is defined identically to that of the storm in **Around The Storm**. The entire ***storm*** is defined as $M_0||M_1||\cdots||M_{22}$.

The behavior of the ***aircraft*** will be modeled by an MDP, M_p as follows:
The parallel composition required to model the *storm* alone contains 2^{22} states; the final parallel composition that models the joint behavior of the *storm* with the *aircraft* contains 2^{29} states (that's 68,719,476,736 states!). This is far too many states for my resources, so a **state space reduction** is necessary.

In order to reduce the state space of this system, the space available to the *aircraft* will be reduced to a 5x5 grid with 9 *storm cells* as follows:



With unchanged behavior for the *aircraft* and each *storm cell*, the state space is reduced to $2^9 \times 50$ states (down to 25,600).