This folder contains PRISM models and a semantic interpreter for a language $\mathcal{L}_{QP}$ described in the paper

"Quantitative Programming and Markov Decision Processes"

Author: Eneia Nicolae Todoran (SYNASC 2022 submission 41)

The paper introduces an experimental concurrent programming language $\mathcal{L}_{QP}$, designed to facilitate the construction of models that capture the behavior of programs and that can be verified formally. Concurrent $\mathcal{L}_{QP}$ programs are translated into corresponding probabilistic models, which are analyzed using the PRISM probabilistic model checker. PRISM is a widely used probabilitic model checking tool (`www.prismmodelchecker.org`).

For formal verification, in this paper we use Markov Decision Processes (MDPs). The paper offers an $\mathcal{L}_{QP}$ program presented in Section III, which is analyzed formally in Section IV.

- The file `epollingsystem-mdp.prism` contains the PRISM MDP model described in Section IV. The experiments presented in Section IV can be performed by using the PRISM model contained in the file `epollingsystem-mdp.prism` and the properties contained in the file `epollingsystem-mdp.props`.
- These experiments are briefly described below (further explanations regarding the meaning of the PRISM variable names employed below are provided in the paper). The variables `nmax` and `wfile` are the PRISM counterparts of the constant $\overline{\mu}$ and variable w_{file}, respectively, described in Section III.
- The approach presented in the paper enables the formal verification of programs with large state spaces by partitioning the state space of programs into into bisimulation equivalence classes.
- The $\mathcal{L}_{QP}$ example program under verification has more than $\overline{\mu}!$ ($\overline{\mu}$ factorial) states.
- In the PRISM experiments presented below we put $\overline{\mu} = 1000$. Some experiments require a considerable amount of time.[1]

[1] On a processor Intel(R) Core(TM) i5-7200U with CPU 2.50 GHz, PRISM completed the experiment presented in Figure 3 in more than 5 hours. The experiments presented in Figure 1 and Figure 2 completed much faster (in 7 minutes and less than 2 minutes, respectively).

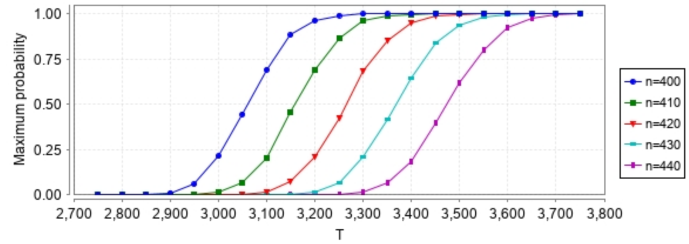

Figure 1: The maximum probability of w_{file} size eventually being equal to `n` within `T` time units

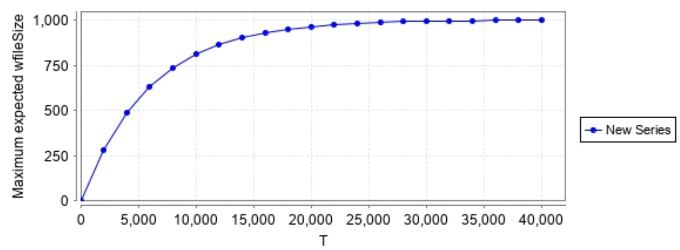

Figure 2: The maximum expected w_{file} size at time instant `T`

- We use the following constants:

```
const int T;
const int n;
const nmax = 1000;
```

1. Using the PRISM property specification language, the maximum probability of w_{file} size (length) eventually being equal to `n` within `T` time units can be specified by:

```
Pmax=? [ F<=T  (vwfile = n)]
```

   Figure 1 presents a PRISM experiment for this property. In this experiment `n` ranges from `400` to `440` with step `10`, and `T` ranges from `2750` to `3750` with step `50`.

2. Using the rewards structure

```
rewards "wfileSize"
  true : vwfile;
endrewards
```

   We can compute the maximum expected w_{file} size at time instant `T` as follows:

```
R{"wfileSize"}max=? [I=T]
```

   Figure 2 presents a PRISM experiment where `T` ranges from `0` to `40000` with step `2000` (`n` is fixed).

   By using the PRISM property specification language, the maximum transient probability that w_{file} contains $\overline{\mu}$ elements at time instant `T` can be specified as follows:

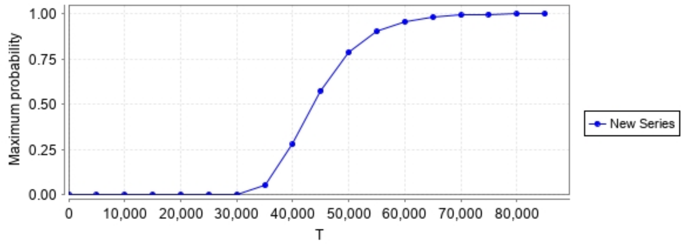

Figure 3: The maximum (transient) probability that w_{file} contains $\overline{\mu}$ elements at time instant `T`

```
Pmax=? [F[T,T] vwfile = nmax]
```

Figure 3 shows a PRISM experiment based on this property, where `T` ranges from `0` to `85000` with step `5000` (`n` is fixed).

- The file `Lqp.hs` contains the semantic interpreter for the language $\mathcal{L}_{QP}$. The semantic interpreter for $\mathcal{L}_{QP}$ is implemented in Haskell (`www.haskell.org`). The language $\mathcal{L}_{QP}$ is described in Section III of the paper.
  - The semantic interpreter can be used to run the $\mathcal{L}_{QP}$ program presented in Section III (further explanations are provided as comments in file `Lqp.hs`).