Probabilistic Model Checking of Systems with a Large State Space: A Stratified Approach

Shou-pon Lin and Nicholas Maxemchuk

NSF project 1329593

Columbia University in the City of New York

Contributions

- While most methods that cope with state explosion problem aim at reducing the problem size, we attack the problem by directed state traversal -- **prioritizing the more probable states in state traversal**
- If complete state traversal is not possible due to limited memory, we may compute an upper-bound of probability for reaching the acceptance state

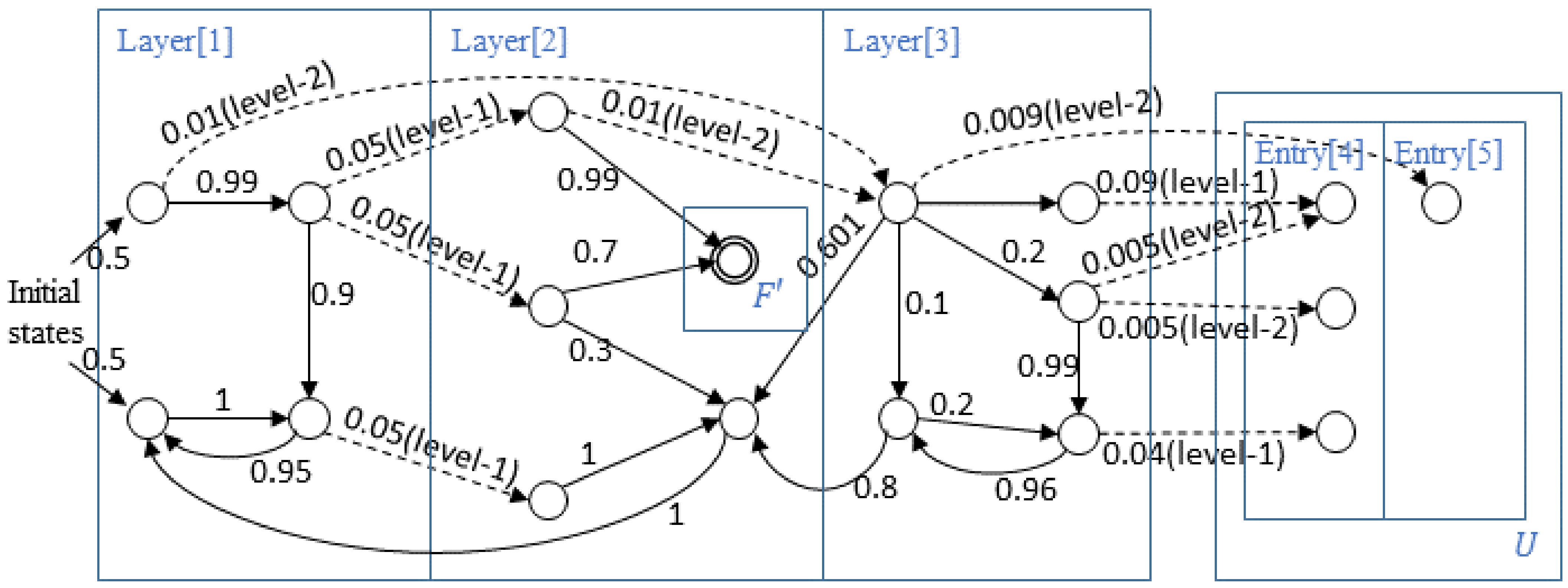
Probabilistic safety property

- **We check if an MDP M satisfies a given probabilistic safety property $\langle A \rangle_{\geq p}$**
  - *where A is a regular safety property and p is a probability bound*
  - *M satisfies $\langle A \rangle_{\geq p}$ if the probability of satisfying A is at least p for any adversary σ*

$$M \models \langle A \rangle_{\geq p} \Leftrightarrow \forall \sigma \in Adv_M \cdot Pr^{\sigma}_{M}(A) \geq p$$
$$\Leftrightarrow Pr^{\min}_{M}(A) \geq p$$

Dividing a Markov Decision Process into Layers

- **Given a layering parameter $\hat{p}$, probabilistic choices are categorized into several discretization levels:**
  1. *(s,α,t) is a (level-0) high probability transition if $P(s,\alpha,t) > \hat{p}$*
  2. *(s,α,t) is a level-1 low probability transition if $\hat{p} \geq P(s,\alpha,t) > \hat{p}^2$*
  3. *(s,α,t) is a level-2 low probability transition if $\hat{p}^2 \geq P(s,\alpha,t) > \hat{p}^3$*
  4. *and so on..*
- **A reachable state *s* belong to layer *k* if *k* is the minimum possible sum of transition levels on any path that reach *s***



Stratified State Traversal Algorithm

Algorithm 1 Stratified Verification of MDP

```
procedure STRATIFIED-DFS(M', p̂)
    Entry[1] ← {s ∈ S | η_init(s) > 0}
    k ← 1
    while ∃i ≥ k s.t. Entry[i] ≠ φ do
        for all s ∈ Entry[k] do
            if s ∉ Layer[i], ∀i ≤ k then
                Insert s into Layer[k]
                STRATIFIED-DFS-VISIT(M', p̂, s, k)
            end if
        end for
        k ← k + 1
    end while
end procedure
procedure STRATIFIED-DFS-VISIT(M', p̂, s, k)
    if s ∈ F then
        Insert s into F'
    end if
    for all (s, α, t) ∈ trans(s) do
        if P(s, α, t) > p̂ then              ▷ high prob. transition
            if t ∉ Layer[i], ∀i ≤ k and t ∉ I then
                Insert t into Layer[k]
                STRATIFIED-DFS-VISIT(M', t, k)
            end if
        else                                 ▷ low prob. transition
            Insert t into Entry[k + ⌊log_p̂ P(s, α, t)⌋]
        end if
    end for
end procedure
```

Check if the MDP satisfied probabilistic safety property

- **Given a set of states F', we compute $Pr^{\min}_{M}(A) = 1 - Pr^{\max}_{M \otimes A^{err}}(\Diamond F')$ by solving a linear program**
- **Suppose the procedure stops at iteration k**

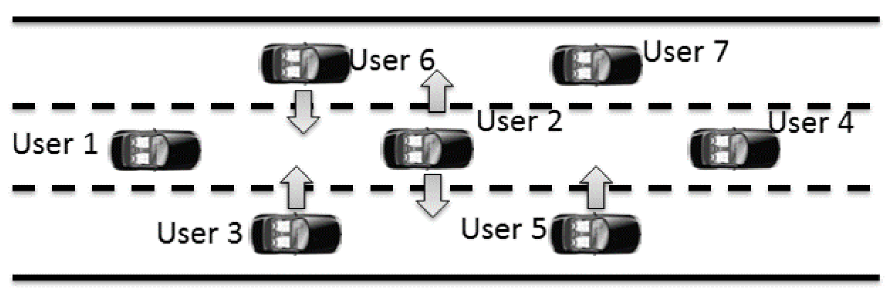
  1.If $1 - Pr^{\max}_{M'}(\Diamond F' \vee \Diamond U) \geq p$, $\langle A \rangle_{\geq p}$ *holds* $(M' = M \otimes A^{err})$

  2.If $1 - Pr^{\max}_{M'}(\Diamond F') < p$, $\langle A \rangle_{\geq p}$ *is violated*

  3.Otherwise, whether $\langle A \rangle_{\geq p}$ *holds or not is uncertain*

Results

- We use stratified verification to consider the lock protocol in [1]. It is applied to a 7-vehicle scenario in which there are 5 conflicting merge requests



- Stratified verification is compared with the explicit engine of PRISM under limited memory constraints. Preliminary results show that stratified verification is able to compute the upper-bound of error probability while PRISM terminates when running out of memory

	Lock with 5 conflicting reqs	
memory budget	PRISM (explicit)	Stratified
75MB	out of memory	4.00312×10^{-4}
100MB	out of memory	4.06118×10^{-13}
150MB	out of memory	9.83204×10^{-18}

- [1]: Shou-pon Lin and Nicholas F Maxemchuk. The fail-safe operation of collaborative driving systems. Journal of Intelligent Transportation Systems, (ahead-of-print), pp. 1-14, 2014.