Verifying Time Partitioning in the DEOS Scheduling Kernel

John Penix, Willem Visser, Corina Pasareanu
NASA Ames Research Center

Eric Engstrom, Aaron Larson, Nicolas Weininger
Honeywell Technology Center

May 5, 2008

Motivace

- Důsledné testování software je drahé a nespolehlivé,
- důležitost software v průmyslu vzrůstá.

- Nejlepší známé metody pro hledání chyb v implementaci:
- **testování** – malé pokrytí, zvláště u souběžných systémů,
- **statická analýza** – obecně náročná.

Úvod

- **Nápad:** model checking k hledání implementačních chyb
- (tradičněji užíván ve fázi návrhu systému).

- **Case Study:** Část DEOS kernelu přepsána do Promely,
- vytvořen model prostředí – klíčový pro úspěch projektu.
- Model příliš velký: nutná abstrakce.

- Hledání známého problému v implementaci.

ParaDiSe
Parallel & Distributed Systems Laboratory

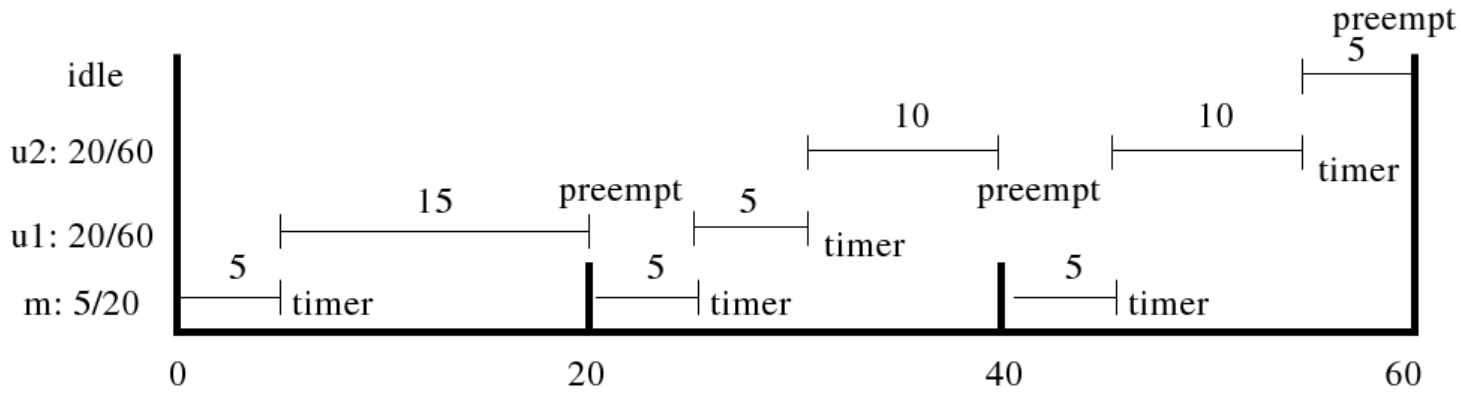
DEOS

- Mikrojádrový operační systém s real-time zárukami,
- přiděluje prostor a čas procesům a vláknům.
- Umožňuje aplikacím různých priorit koexistovat na společném fyzickém zařízení.
- Garantuje neinterferenci mezi aplikacemi $\longrightarrow$ méně interakce usnadňuje ověřování jednotlivých aplikací.

ParaDiSe
Parallel & Distributed Systems Laboratory

Modelování DEOSu

- Time partitioning property: každé vlákno má zaručeno, že dostane celý svůj časový rozpočet v každé periodě.



Modelování DEOSu: *Čas*

- Diskretizované hodiny: *Variable time advance*,
- při každé události se uhodne/spočte další.
- Tick: pravidelné hodiny s fixním intervalem.
- Timer: stopky, pokaždé se nastaví nový čas.

- Explicitně namodelovaný časovač: 2 člověkoměsíce práce,
- asi stránka kódu v Promele.

ParaDiSe
Parallel & Distributed Systems Laboratory

Modelování DEOSu: *Rozhraní časovače*

- **start** : spustí časovač na *n* časových jednotek
- **timer interrupt** : časovač doběhl
- **tick** : konec nejrychlejší periody
- **remaining** : dotaz na zbývající čas běžícího vlákna
  abstrakce: vrací pouze 0, maximum, maximum/2.

ParaDiSe
Parallel & Distributed Systems Laboratory

Modelování DEOSu: *Čas – LTL omezení*

- Univerzální model: obsahuje chování která neodpovídají chování skutečného časovače $\rightarrow$ množství falešných protipříkladů (ale **remaining** vrací pouze 0 nebo maximum...
- Idea: omezit chování pomocí vhodných LTL vlastností.
- ϕ_1: Hodiny lze resetovat jen pokud je jejich hodnota 0: $\mathbf{G}(tick \rightarrow \neg newtick\ \mathbf{U}\ (clock = 0 \vee \mathbf{G} \neg newtick))$
- ϕ_2: Zbývající čas vlákna v momentě přerušení je 0: $\mathbf{G}(timer \rightarrow \neg remaining \geq 0\ \mathbf{U}\ (start \vee \mathbf{G}\ \neg remaining \geq 0))$
- Model U_TIMER už splňuje ϕ_1, omezí se pouze pomocí ϕ_2.
- Jak implementovat? Syntéza $\vee$ předpoklad v neverclaimu.

ParaDiSe Parallel & Distributed Systems Laboratory

Abstrakce

- Cíl: snížit náročnost verifikace (zmenšit stavový prostor při zachování zajímavých vlastností).
- Zachování vlastnosti:
  - **silné** : $\forall \phi, M \models \phi.\ M' \models \phi$ – zajímavé jen pro malou množinu vlastností
  - **slabé** : $\forall \phi, M \models \phi.\ \exists \phi', M' \models \phi'$ – zásadnější redukce.
- Vliv na možná chování systému:
  - **over**-aproximace: přidání „nemožných“ chování,
  - **under**-aproximace: odstranění realistických chování.

ParaDiSe
Parallel & Distributed Systems Laboratory

Predikátová abstrakce

- Myšlenka: nahradit proměnnou nebo sadu proměnných binárním predikátem.
- Over-aproximace, při manipulaci s proměnnými je často nutné nedeterministicky se rozhodnout jak manipulace změní pravdivost predikátu.
- Příklad: x, y počítadla, zajímá nás pouze zda $x = y$.
  - Zavedeme $B \equiv x = y$,
  - $x++$ přepíšeme na:
    $B = \text{false} \rightarrow B = \text{true} \;||\; B = \text{false}$,
    $B = \text{true} \rightarrow B = \text{false}$,
  - y analogicky.

ParaDiSe Parallel & Distributed Systems Laboratory

Diskuse a/nebo abstrakce a OOP.

ParaDiSe
Parallel & Distributed Systems Laboratory