Modelování řízení souběžného přístupu k datům v real-time databázích

Martin Kot

Katedra informatiky
VŠB - Technická univerzita Ostrava

5. února 2008

Modelování a verifikace real-time databázových systémů

- Korektnost systému je možné zjišťovat různými metodami:
  - Testování
  - Simulace
  - Formální verifikace
    - Dokazování vět (theorem proving)
    - Ověřování ekvivalenci (ekvivalence checking)
    - Ověřování modelů (model checking)
- Pro formální verifikaci bylo vytvořeno mnoho softwarových nástrojů
- Snažíme se využít Uppaal na verifikaci real-time databázového systému

Uppaal

- Softwarový nástroj pro automatizovanou verifikaci real-time systémů
- Systém je modelován jako síť časovaných automatů rozšířených o další možnosti
- Časovaný automat je konečně stavový automat rozšířený o časové proměnné
- Automaty v síti se synchronizují a předávají si hodnoty
- Stav systému je dán aktuálními lokacemi všech automatů a hodnotou všech hodin a diskrétních proměnných
- Způsoby změny stavu:
  - plynutí času
  - provedení hrany některého z automatů

Uppaal

- Uppaal umožňuje:
  - provádět simulaci na modelech
  - ověřovat vlastnosti zadané pomocí temporální logiky
- Základní vlastnosti, které je možné ověřovat:
  - reachability – „Je dosažitelný stav, kdy je něco dobrého splněno?“
  - safety – „Je něco dobrého vždy splněno?“
  - liveness – „Nastane něco dobrého nakonec vždy“ nebo „Vede splnění něčeho vždy k něčemu jinému?“

Real-time databázové systémy

Společné vlastnosti s konvenčními databázovými systémy

- ukládání dat
- efektivní algoritmy pro ukládání, vyhledávání a manipulaci s daty

Specifické vlastnosti real-time databázových systémů

- měla by existovat nějaká spolehlivost časové odezvy na požadavek

Použití

- real-time řídící systémy - výrobní linky, telekomunikační systémy, řízení letového provozu
- informační systémy - banky, burzy

Výzkum v oblasti RT databázových systémů

- od 90tých let 20. století
- hlavně zaměřen na algoritmy pro nejdůležitější části
  - řízení souběžného přístupu
  - indexace dat
  - bufferování dat
  - ...
- navrhované algoritmy
  - adaptované z konvenčních databází
  - zcela nové
  - ...
- možné propojení s real-time operačními systémy
- cíl - omezení času reakce na požadavek

V4DB

- experimentální real-time databázový systém
- navržen a implementován na Katedře měřící a řídící techniky Technické univerzity v Ostravě
- složen ze všech důležitých funkčních částí skutečného real-time databázového systému
- možnost měnit algoritmy jednotlivých částí k pochopení jejich vlivu na chování systému
- centralizovaný, databáze v paměti, navržen pro jeden procesor
- naprogramován pro VxWorks
- stále se vyvíjí a doplňuje

Transakční zpracování

- aplikace nepřistupují přímo k záznamům
- komunikují prostřednictvím databázového řídícího systému
- požadavky jsou často v blocích nazývaných transakce
- databáze může být během zpracování transakce nekonzistentní

Přiřazování priorit

- transakce získávají čas CPU podle priority
- transakce mají různé vlastnosti, které ovlivňují přiřazování priorit:
  - deadline
  - kritičnost
  - množství zbývající aktivity
  - množství již využitého procesorového času
  - trvání transakce
- funkce přiřazující prioritu by měly brát do úvahy více z těchto vlastností
- nejčastěji se používají deadline a kritičnost

Přiřazování priorit

- Deadline-monotonic - jen (relativní) deadline
- Rate-monotonic - pro periodické transakce
- Execute deadline first (EDF) - absolutní deadline
- Adaptive earliest deadline (AED) - EDF s kontrolou zahlcení
- Adaptive earliest virtual deadline (AEVD) - upravena verze AED

Řízení souběžného přístupu

- má zajistit bezchybné paralelní vykonávání transakcí
- paralelismus je vhodný pro čtení
- jen jeden proces by měl zapisovat do dané části databáze nebo do daného záznamu
- sériovost transakcí - výsledek paralelního vykonávání je stejný, jako výsledek některého z možných sériových

Možné problémy paralelního vykonávání:

- ztáta změn
- nekonzistence čtení
- čtení špinavých dat

Řízení souběžného přístupu

Pro databázové řídící systémy bylo navrženo mnoho protokolů řízení souběžného přístupu:

- pesimistické
- optimistické
- spekulativní
- ...

V případě RT databázových systému by měly být přidány deadline, priority atd. Používají se:

- pesimistické
- optimistické

Pesimistické protokoly

- založeny na zámcích
- dva konfliktní módy - čtecí a zapisovací
- zámek - proměnná přiřazená k záznamu, obsahuje typ operace
- samotné zamykaní nezajistí sériovost
- je potřebný vhodný zamykací protokol

Dvoufázové zamykání (2-PL)

- 1. fáze - transakce získává zámky
- 2. fáze - transakce uvolňuje zámky
- typ zámku může být změněn během 1. fáze
- možné problémy
  - inverze priorit
  - deadlock

Modifikace 2-PL

- 2-PL wait promote - dědičnost priorit
- 2-PL high priority - všechny konflikty se řeší ve prospěch transakce s vyšší prioritou
- 2-PL without cyclic restart

Optimistické protokoly

- validace transakcí na konci jejich zpracování
- žádné řízení během zpracování
- typy validace:
  - dopředná - s transakcemi, které se teprve zpracovávají
  - zpětná - s již potvrzenými transakcemi
- konktétní protokoly:
  - OCC broadcast commit - transakce ve validační fázi restartuje všechny konfliktní transakce, které se zpracovávají
  - OCC sacrifice - modifikace OCC-BC, restartována je transakce s menší prioritou
  - OCC wait - modifikace OCC-BC, transakce s menší prioritou čeká

Struktura V4DB

- Generator transakcí – náhodně generuje transakce
- Predispatcher – zabraňuje přetížení systému
- Dispatcher – extrahuje parametry transakcí, předává transakce k zpracování podle priority
- Výkonná jednotka – transakce je rozložena na konkrétní příkazy, které jsou vykonány
- Řízení souběžného přístupu – řídí souběžný přístup více transakcí k databázi

Verifikace V4DB v Uppaalu

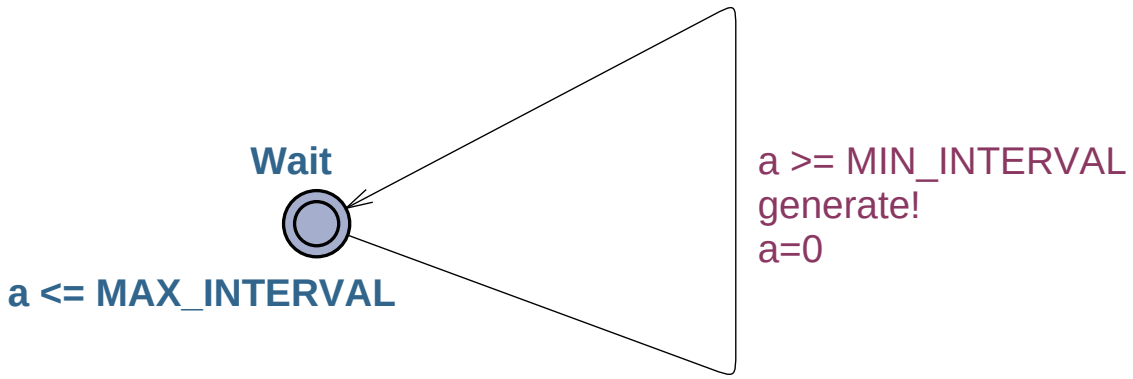
- modelování různých protokolů a algoritmů používaných v V4DB
- identifikace důležitých vlastností a jejich ověření
- snaha pomoci autorům V4DB najít možné chyby a vylepšit jejich systém
- zatím jsme se zaměřili na jednu z nejdůležitějších částí - řízení souběžného přístupu
- byly vytvořeny modely několika různých protokolů souběžného řízení

Model protokolu sacrifice

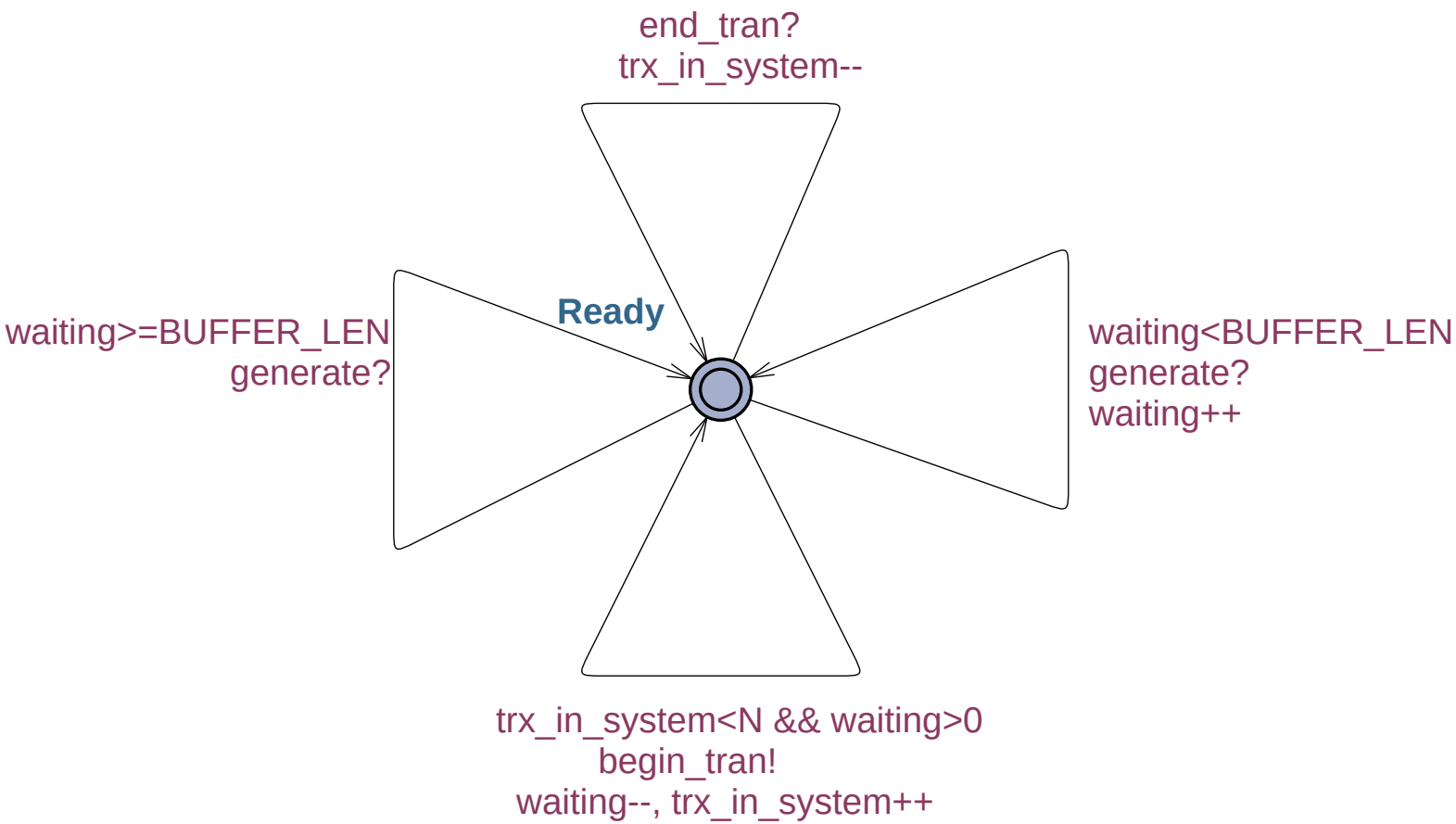
Je nutné najít vhodnou abstrakci systému

- Dispatcher pustí do systému jen omezený počet transakcí současně – každá aktivní transakce je reprezentována jedním automatem
- Protokol řídí přístup k záznamům, kokrétní data nejsou pro jeho činnost důležitá
- Z hlediska řídícího systému je konflikt vlastně náhodná situace
- Databázové operace můžeme modelovat jako časové prodlení
- Priorita transakcí je také do jisté míry náhodná – transakční automaty jsou vybírány nedeterministicky, každý má pevně dánu prioritu

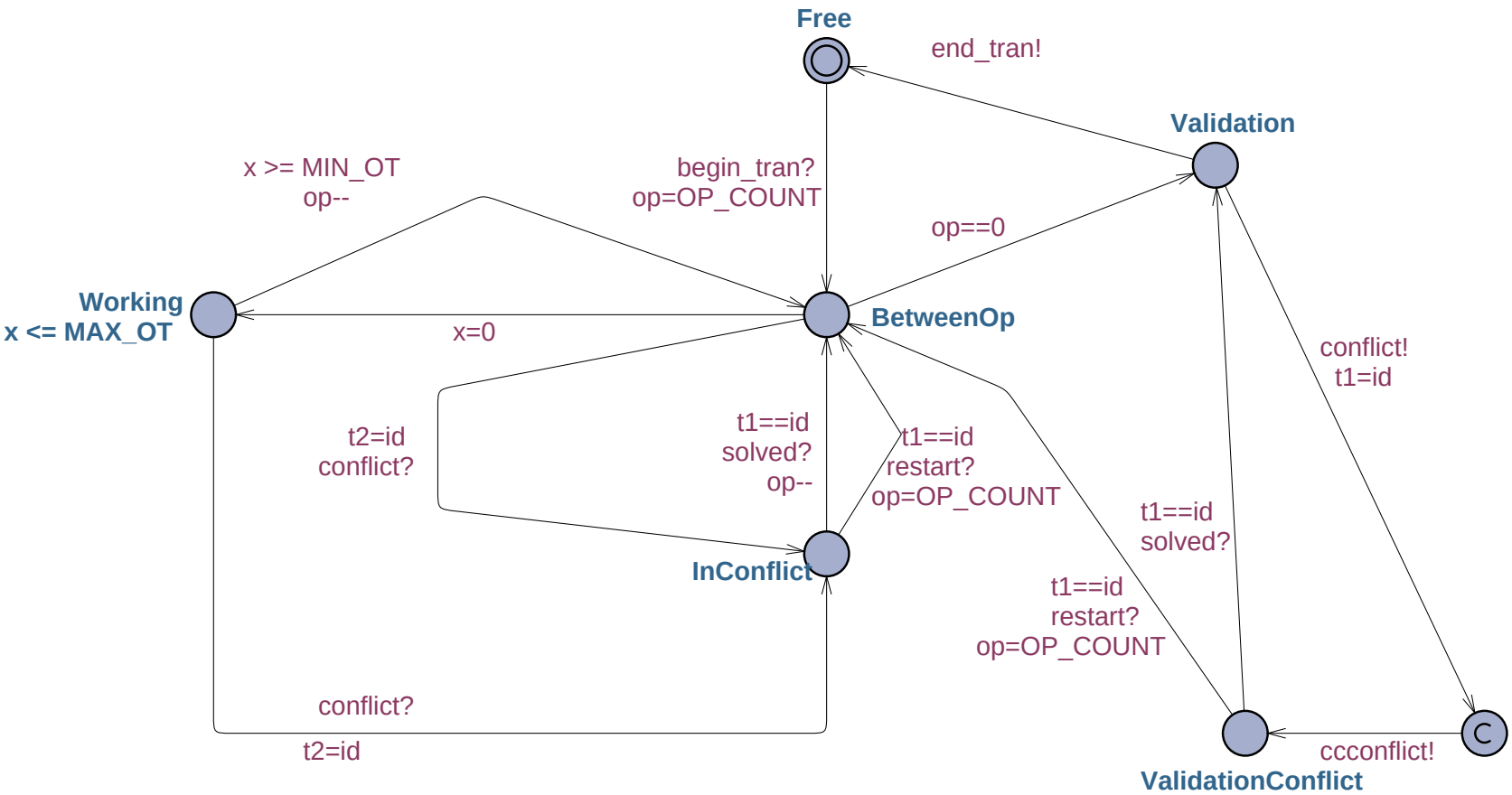
Generátor



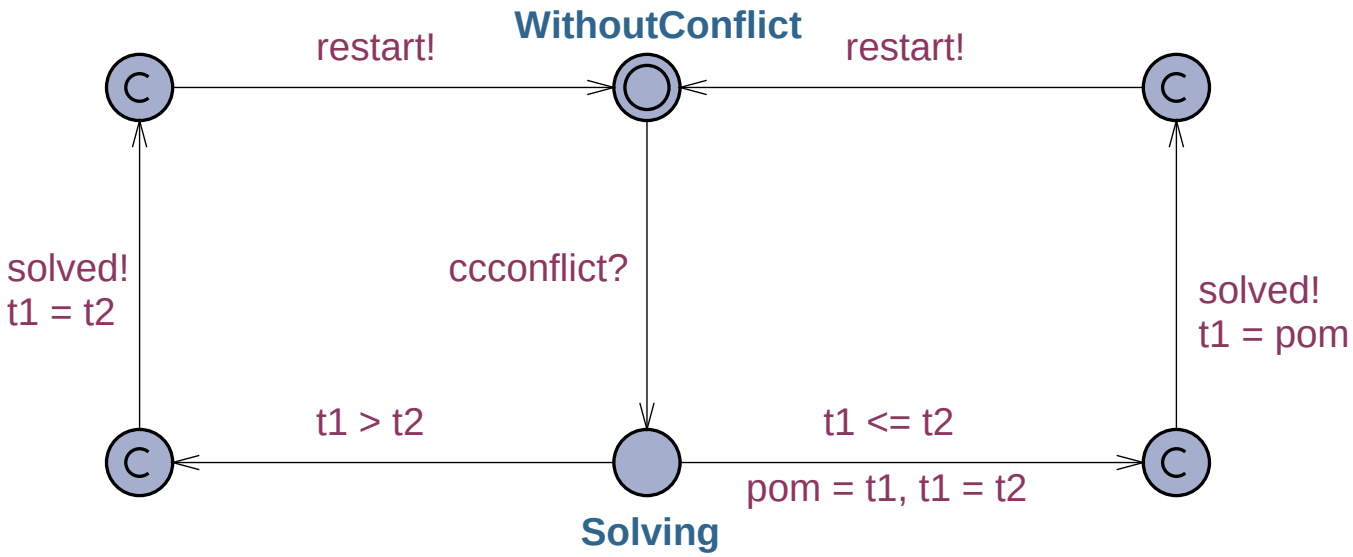
Dispatcher


end_tran?
trx_in_system--
Ready
waiting>=BUFFER_LEN
generate?
waiting<BUFFER_LEN
generate?
waiting++
trx_in_system<N && waiting>0
begin_tran!
waiting--, trx_in_system++

Transakce


Free
end_tran!
Validation
x >= MIN_OT
op--
begin_tran?
op=OP_COUNT
op==0
Working
x <= MAX_OT
x=0
BetweenOp
conflict!
t1=id
t2=id
conflict?
t1==id
solved?
op--
t1==id
restart?
op=OP_COUNT
t1==id
solved?
InConflict
t1==id
restart?
op=OP_COUNT
conflict?
t2=id
ccconflict!
C
ValidationConflict

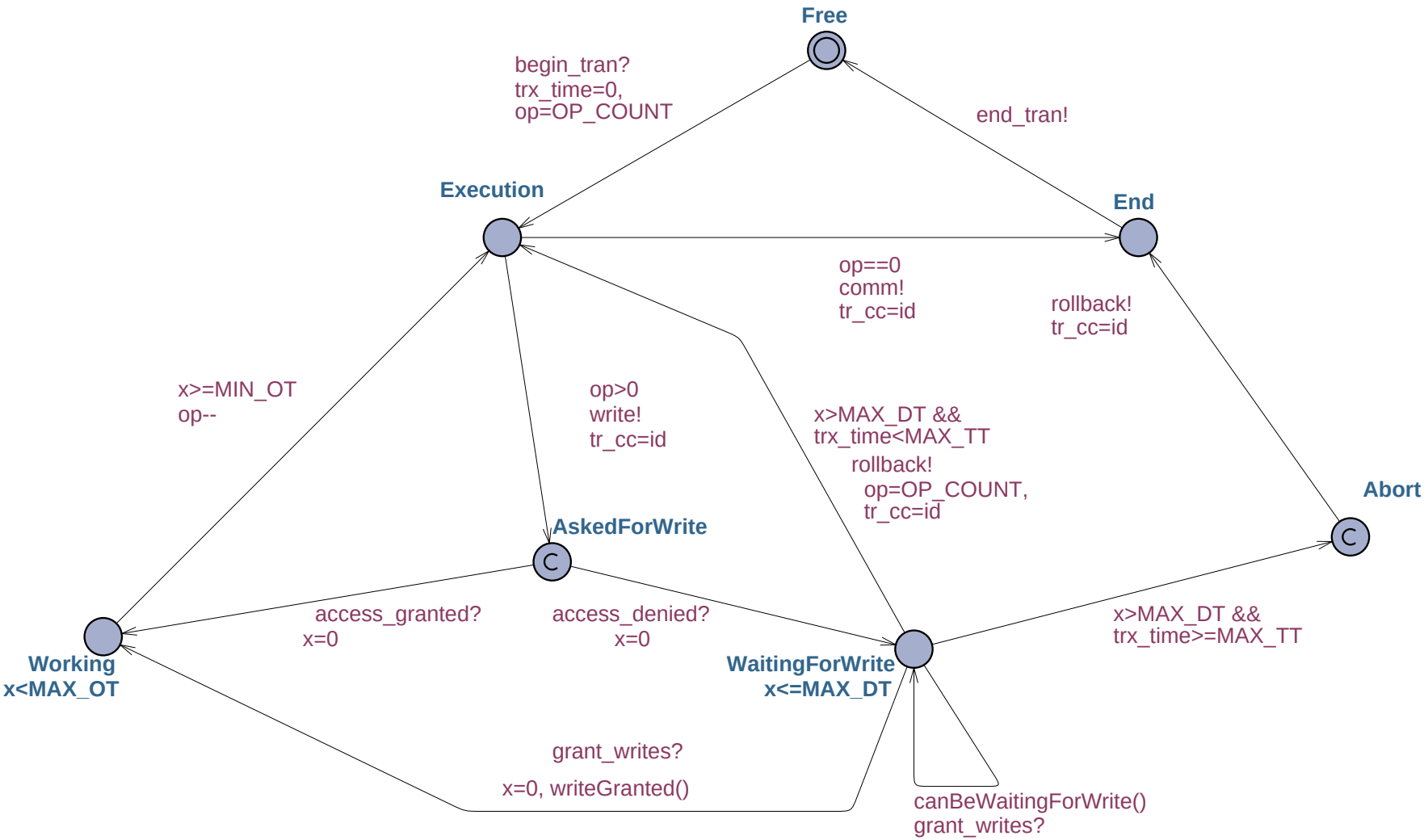
Řízení souběžného přístupu



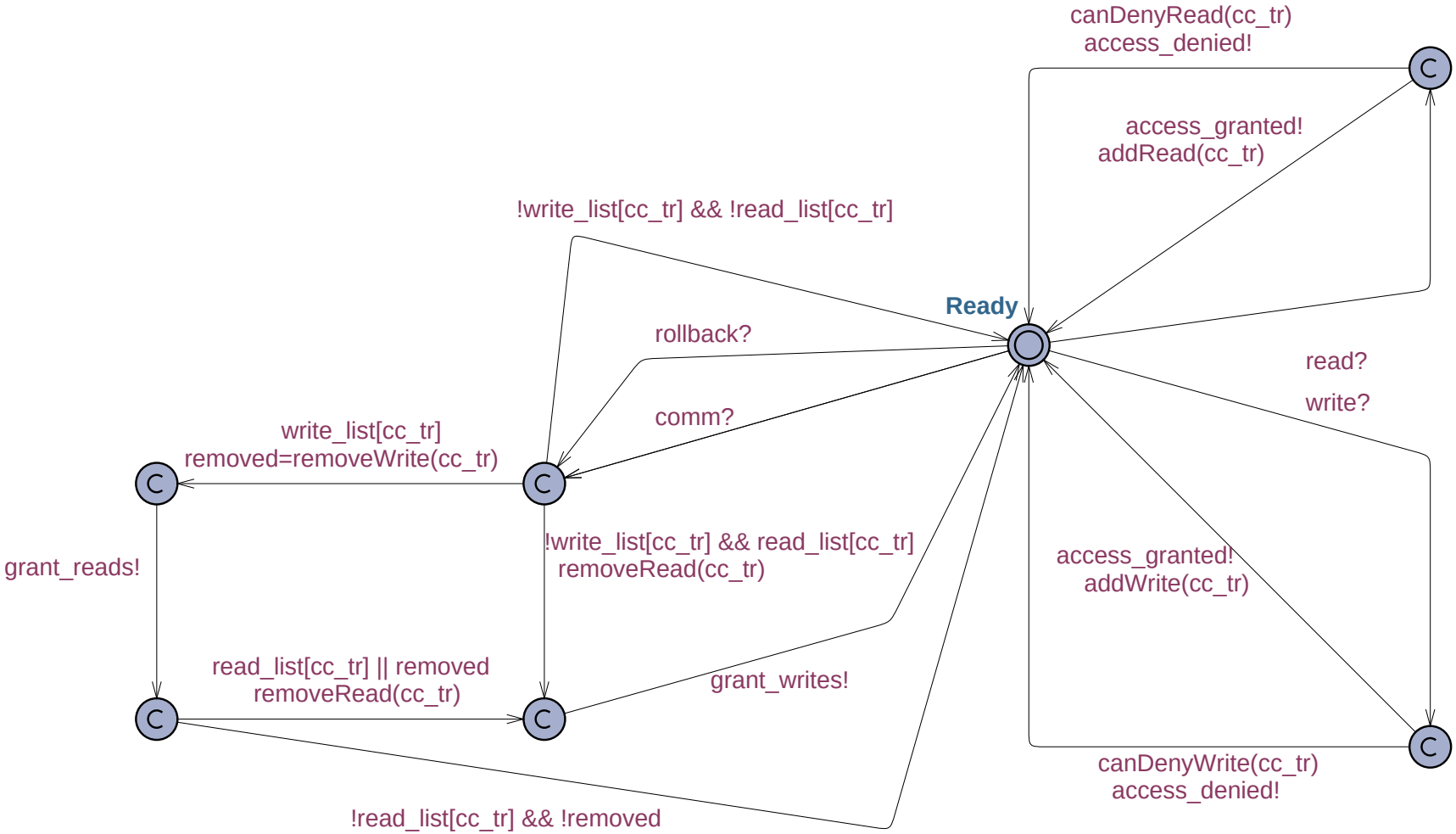
Pesimistický protokol 2-PL

- automaty reprezentující generátor a dispatcher jsou stejné, jako v předchozím případě
- transakční automaty a automat řízení souběžného přístupu musí podchytit informace o zámcích
- není úplně důležité, na kterém záznamu je který typ zámku
- musíme vědět, jestli existuje transakce držící zámky - tehdy jiná může být blokována
- pro jednoduchost modelu, každé uvolnění zámku může odblokovat libovolnou čekající transakci

Transakce


Free
begin_tran?
trx_time=0,
op=OP_COUNT
end_tran!
Execution
End
op==0
comm!
tr_cc=id
rollback!
tr_cc=id
x>=MIN_OT
op--
op>0
write!
tr_cc=id
x>MAX_DT &&
trx_time<MAX_TT
rollback!
op=OP_COUNT,
tr_cc=id
Abort
AskedForWrite
C
C
access_granted?
x=0
access_denied?
x=0
x>MAX_DT &&
trx_time>=MAX_TT
Working
x<MAX_OT
WaitingForWrite
x<=MAX_DT
grant_writes?
x=0, writeGranted()
canBeWaitingForWrite()
grant_writes?

Řízení souběžného přístupu



Verifikace

- pro podchycení zajímavých vlastností souběžného přístupu k datům potřebujeme několik automatů reprezentujících transakce
- roste stavový prostor - tzv. state-space explosion
- i ověření jednoduchých dotazů selhává na nedostatek paměti
- postupně se modely upravují, zjednodušují apod. aby se dařilo ověřovat více vlastností
- dále je nutné identifikovat skutečně zajímavé vlastnosti pro autory V4DB a ty ověřit

Budoucí práce

- další vylepšování existujících modelů a ověřování jejich vlastností
- modelování dalších částí RT databázového systému
- využití nových možností připravované verze Uppaalu a případně i jiných verifikačních nástrojů