abstract regular model checking

about

original:(https://www.fit.vutbr.cz/~vojnar/Publications/armc-sttt-12.pdf)

概要

regular model checking 常用於檢查
1. 無限狀態
2. parametrised systems
   - 使用有限 word/tree automata 表示潛在的無限可到達集合

透過正規模型檢查，這個問題是典型的不可決定問題 爲了盡可能的促進中止，需要加速可到達配置的計算 在這個論文中， 我們敘述各式各樣的逐步提煉的抽象方法 用在有限 word/tree automata 上 來達到加速

抽象的用途不只是增加中止的機會 他也可以簡化 regular model checking 產生 automata 爆炸多的狀態數量

我們演示 已經確認過的無限資源簡單系統（像是無界計數器，queue,stacks,parameter） 來實作抽象正規模型檢查

然後我們展示用抽象正規樹的模型檢查 可以用在確認操作樹狀資料結構的程式

介紹

確認有限狀態系統裏，模型檢查是現在常用的有力科技 然而很多現實系統有各種無限狀態

我們在論文中集中一些離散系統 無限可能會出現在可以處裏各種無界限資料結構中
- 像是 push-down stacks 需要處裏遞迴程式
- queue 需要等待 process/message
- 無限計數器，動態連接的資料結構

需要處裏無限狀態也會讓需要的儲存空間變多

最後 爲了更精確我們要處裏 infinite families of systems
- 有限狀態
- 無限狀態

儘管如此 因爲所有家族成員狀態空間的 union 是無限的 所以 需要確認系統 出現哪個家族成員導致無限的狀態驗證 > 出現家族成員會導致無限狀態驗證 > 需要在系統裏指出無限狀態的存在

爲了處裏無限狀態
- 嘗試找出無限資源的有限邊界（AKA cut-offs）
- 可以使用各種有限界限 抽象 或是 自動機簡化科技
- symbolic model checking 基於用有限表示 等價 無限狀態的集合（邏輯等價，自動機等價, grammars 等價）

在成功的代數檢查法中我們也有正規（樹）模型檢查-R(T)MC 首次出現於[36]論文中

在 R(T)MC 中，系統配置加密成有限字母的 word 或 tree 然而 Transitions 被改成有限狀態的 transducers 或是存成 word 或是 tree 有限自動機可以自然的代表或是操作淺在地無限配置集合 讓 transducers 的傳遞閉包或是 transducers 靠着 reachability 關係迭代出的圖像 automata 計算可到達的特性

爲了促進計算中止，常用各種加速方法，由於是不可決定的問題，通常不能保證計算中止

在這篇論文中，我們提出了一系列的抽象方法來加速。 我們要用 R(T)MC 加讓 CEGAR loop 來達成目標

我們在有限的自動機領域用抽象的不定點計算 而不是使用 精確的加速科技 抽象的不動點計算總是會中止然後提供非常近似的 relations

爲了達成這個目標， 我們有系統的映射任何自動機 M 到一個抽象自動機 M'的科技 M'的 domain 是有限的 而且 M'可以辨認 M 的 superset

如果計算過的過近似太粗操而且偵測到虛假反例 我們提供有效率的技術 讓抽象精煉時不會出現一樣的虛假反例

兩種抽象方法
我們討論字串和樹兩種常見目標種類的抽象自動機技術 他們有考慮到自動機的結構 而且基於等價的方式摺疊這些狀態

字串是受到 predicate abstraction 的啓發 然而跟典型的 predicate abstraction 不同 我們指定 predicate 有設置 狀態代表自動機 而不是他們自己的設置

抽象是正規斷言(predicate) languageL_p集合定義的 如果L_p交集$L(M,q)$中的 q 不是空集 稱爲狀態 q 在 M 裏滿足L_p

兩個狀態滿足一樣的斷定 =>兩個狀態相等 基於 如果兩個自動機狀態的語言達到一定的相同固定長度時 認爲相等

對於這兩個方法我們提供有效率的細化技術讓我們可以移除虛假反例

上面說的技術已經被實作在 prototype 工具上，而且被用在各種測試上 尤其是抽象正規字模型檢查中特別成功 應用在確認程序的參數網路，pushdown 系統，計數器自動機，queue 系統，動態單連結構程式

先備知識

有限字串自動機和 **transducers**
有限狀態自動機敘述爲 $M = (Q, \Sigma, \delta, q_0, F)$
- Q 是有限狀態集合
- Σ 是有限字母集合
- δ:$Q \times \Sigma \to 2^Q$ 是 transition 函數
- q_0是初始狀態
- F 是接受狀態集合

轉換關西 $\underset{M}{\longrightarrow}$ 定義爲 $\subseteq Q \times \Sigma^* \times Q$ of M 最小的關係滿足
1. $\forall q \in Q : q \xrightarrow[M]{w} q'$
2. if $q' \in \delta(q,a)$ then $q \xrightarrow[M]{a} q'$
3. if $q \xrightarrow[M]{w} q'$ and $q' \xrightarrow[M]{a} a''$ then $q \xrightarrow[M]{\{wa\}} q''$

for $a \in \Sigma, w \in \Sigma^*$ 如果 M 有可能沒有混亂的狀態那就捨棄 M M 也叫可決定的 iff$\forall q \in Q \forall a \in \Sigma : |\delta(q,a)| \leq 1$

從狀態 q 開始的有限狀態自動機 M 構成的語言標記爲 $L(M,q) = \left\{ w \in \Sigma^* \mid \exists q_F \in F : q \underset{M}{\longrightarrow} * wq_F \right\}$ 語言 L(M)是$L(M,q_0)$的簡寫

集合$L \subseteq \Sigma^*$ 是正規集合 iff 存在有限狀態自動機 M 滿足$L = L(M)$

定義 backward/forward 語言 $L(M,q) = \left\{ w \mid q_0 \xrightarrow[M]{w} q \right\}$ w 有特定長度：$L^{\{\leq n\}}(M,q) = \{w \in L(M,q) \mid |w| \leq n\}$ 向前 / 向後的 trace language of states $T(M,q) = \{w \in \Sigma^* \mid \exists w' \in \Sigma^* : ww' \in L(M,q)\}$

最後 定義 accordingly 向前 / 向後的 trace language $T^{\{\leq n\}}(M,q)$ =and $\hat{T}^{\{\leq n\}}(M,q)$

給有限狀態自動機 M=(Q,Σ,δ,q_0,F) 和 等價的關係 在他的狀態集合 Q 上 $M_{/\sim}$表示 quotient automaton of M $M_{/\sim} = \left(Q_{/\sim}, \Sigma, \delta_{/\sim}, [q_0]_{/\sim}, F_{/\sim}\right)$ where $Q_{/\sim}$ and $F_{/\sim}$ are Q 和 F 的參考 的部分 $[q_0]_{/\sim}$是 Q 的參考 的部分而且含有q_0

有限狀態轉少機 $\tau = (Q, \Sigma, \delta, q_0, F)$
- Q 是有限狀態集合
- Σ 是有限輸入輸出字母集合
- δ:$Q \times \Sigma_\varepsilon \times \Sigma_\varepsilon \to 2^Q$ 是 transition 函數，

其中$\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$
- q_0是初始狀態
- F 是接受狀態集合

在轉換不包含

$$\varepsilon$$

的情況下 有限狀態轉少機又被稱作長度保留轉少機

轉換關係 $\xrightarrow[\tau]{} \subseteq Q \times \Sigma^* \times \Sigma^*$ 被定義成最小關係需要滿足：
1. $\forall q \in Q, q \xrightarrow[\tau]{\varepsilon/\varepsilon} q$
2. if $q' \in \delta(q,a,b)$ then $q \xrightarrow[\tau]{a/b} q'$
3. if $q \xrightarrow[\tau]{w/u} q'$ and $q' \xrightarrow[\tau]{a/b} q''$

then $q \xrightarrow[\tau]{wa/ub} q''$ $a, b \in \Sigma_\varepsilon, w, u \in \Sigma^*$

子腳本 τ 再沒有作用的情況會被丟掉

給有限狀態轉換機 $\tau = (Q, \Sigma, \delta, q_0, F)$ 定義關係 $\varrho_\tau = \left\{ (w,u) \in \Sigma^* \times \Sigma^* \mid \exists q \in Q_F \in F : q_0 \xrightarrow[\tau]{w/u} q_F \right\}$

關係 $\rho \subseteq \Sigma^* \times \Sigma^*$ 是正規關係 iff 存在有限狀態轉少機 tau 滿足 $\rho = \varrho_\tau$

對於集合$L \subseteq \Sigma^*$ 和關係 $\rho \subseteq \Sigma^* \times \Sigma^*$ 記爲 $\varrho(L)$ 是 集合$\{w \in \Sigma^* \mid \exists w' \in L : (w', w) \in \varrho\}$

關係 $\rho \subseteq \Sigma^* \times \Sigma^*$ 是正規性保持 iff $\varrho(L)$ 是正規的對於所有正規集合$L \subseteq \Sigma^*$

Note：不是所有正規性保持的關係都是正規的 像是 $\{(w, w^R) \mid w \in \Sigma^*\}$, w^R是 w 的 reverse, $|\Sigma| > 1$

有限樹自動機和 **transducers**
有限字母集 Σ 可以排名 排名 function #: $\Sigma \to \mathbb{N}$ 對於每個 $k \in \mathbb{N}, \Sigma_k \subseteq \Sigma$ 是所有標誌有排行 k 的集合

Σ_0 的標誌叫做 contstants(常數)

χ是標誌的可數集合 叫做 variables(變數) $T_{\{\Sigma\}}[\chi]$ 是所有由 Σ 和 χ集成的術語集 $T_{\{\Sigma\}}[\varphi]$ 標記成 $T_{\{\Sigma\}}$，他的元素叫做 ground terms(基本術語) 如果每個變數最多在 t 中出現一次，$T_{\{\Sigma\}}[\chi]$中的術語 t 叫做線性

一組標籤 L 上的有限有序樹 t 是映射 t[WIP]

正規模型檢查

基礎想法
我們在概論中有說過 基礎想法是考慮系統中的基礎配置 編碼成 適合的有限字母表上的 words 並表示無限但是正規 這些有限狀態自動機設置的集合

在設置之間的轉換構成給定系統的一步轉換關係 然後用有限狀態轉少機來編碼 或是更常見的用正規維持關係表達（像是專門的自動機操作）

在這個部分，我們爲了簡單 一致的使用單一的轉少機編碼 給定系統的一步轉換關係

再進去技術細節裏之前展示一個簡單地例子：線性拓撲程序的參數網路區域認證 當處裏這個系統時，每個字母在字串裏都代表一個進程的狀態 字串的長度代表有幾個進程在系統裏 特別考慮非常簡單的 token 經過協定 任意有限數量進程進入線性網路 每個進程中都沒有 token，要等 token 從左邊進來 有 token 的進程會把 token 傳給右邊的進程 一開始只有最左邊的進程有 token

在協定裏編碼每個進程的狀態 $\Sigma = \{N, T\}$ 足以表達 無 token 和有 token 的狀態 Set I 是可能的初始設置 I 可以被編碼成自動機 Fig 1(a) 轉少機τ裏的一步轉換關係 Fig 1(b)

一旦有一個轉少機編碼一步轉換關係 χ 還有自動機編碼他的初始設置集合 I 就有兩個基礎策略可以使用 也是可以直接嘗試計算所有可以到達的設置集合$\chi^*(I)$ 或是可到達關係χ^*

$\chi^*(I)$可在重複的目前以到達狀態集合的一步轉換關係χ後被維持

這個問題在參數化的上下文和無限狀態系統 如果用向前固定點計算無限的集合，計算不會停

爲了計算$\chi^*(I)$ or χ^* 至少在特別的情況下停止 需要加速計算 一次可以就說出敘述無限數字的結果

在例子中 token 一步一步移動到右邊 一步內允許 token 移動任意距離可以加速定點計算 用這個加速可以馬上得到達 Fig3.a 的固定點，他代表協定中所有可到達設置的集合 文章中的目標是在正規模型檢查裏很多不同方法加速定點計算 3,3 會簡單複習這些方法

3.2 正規模型檢查認證[WIP]

安全性能檢查可以減少檢查不會到達沒有壞的狀態 如果我們要檢查不能到達的壞狀態裏 可以被表達成正規集合 B 我們可以簡單地計算可到達集合 然後檢查他跟 B 的交集是不是空集

像是簡單 token 協定中 考慮沒有 token 或是兩個 token 的狀態會讓系統壞掉 這兩個狀態可以被編碼成有限狀態自動機在 fig3(b) 他很清楚的表達在 fig3(a)中的可到達集合 跟 fig3(b)的交集是空集 所以系統是安全的

用正規模型檢查很難檢查活性特性 世界上的有限狀態系統的活性可以被減少成可到達問題

當研究系統被保持長度的轉移關係模組化 可以正規在模型檢查的前後文中檢查簡單到達

在這個情況下只有一種系統循環方法可以簡單地 重複經過一些設置 剛剛簡單地方法可以用 buchi automata 編碼未決定行爲構建系統然後檢查

note: 上面說的計算中 部長需要計算可到達集合 也要算可到達關係

還有這個步驟要避免猜測 當接受循環開始的時候 把每個設置的 word 中的每個字母都兩倍 然後計算偶數字母當所有基數字母有對應到偶數時透過尋找狀態偵測最近的循環 會在 Section4.5 練習

透過正規模型檢查 參數模型網路程序在有系統框架跟特定優先度 會在[2,3]作爲目標 他用 LTL(MSO)作爲建模語言 表達時間順序邏輯 LTL 表達配置在字元屬性的單元二階邏輯組合 這個方法也目標一個模型的自動機轉化 檢查他們變成自動機框架符合正規模型檢查 計算可到達關係然後用在認證

最後用 不保留長度轉換關係更改系統 檢查 活性屬性 比保持長度轉換更複雜 因爲不保持長度系統會有無限行爲無限的往 buchi 自動機的接受狀態遷移 大概像是無限變大的 queue [17]有說用正規模型檢查自動機計算最大的模擬可到達關係設置是可計算的 伴隨屬性可以被追蹤 然後不指檢查接受設置可以到達 他也檢查自己可以到自己 檢查 c1 可到達通過接受設置 c2 模擬 c1 可到達 學習特別他們產生的樣本目標方式的固定點的替代方法也被作爲目標

3.3 加速正規模型檢查

加速方法設計給正規模型檢查用 包爪加速 schemes, quotienting, extrapolationm inference of regular language 用抽象方法去描述的細節在一愛 section 4 下面會簡單說其他方法

加速 schemes 在[44]中被提出 加速 schemes 允許一個夫生的 mata 轉換加密有隨意次數的轉換影響 [44]有提供三種方法來檢查 [WIP] 我覺得這裏不太重要

4. 抽象正規模型檢查

加速可到達計算可以實際的讓計算中止 另外一個問題是要面臨狀態爆炸的問題 這個問題有關前面提到的正規模型檢查技術

通常這些科技獨立計算計算準確的可到達關係