第一章 緒論

1.1 研究動機

並行機制（Concurrency）提供給軟體系統更多的效能和需求。不幸的是，這使的原本就不易檢驗的系統特性變的更加困難。通常關鍵性的系統(critical systems)可能因為發生系統錯誤而造成無法承受的損失，故檢驗軟體的正確性是對這一類系統是相當重要的。模型檢驗（model checking）能幫助我們發現系統中的錯誤進而修正它。一般來說，並行或分散式的程式由於系統都相當複雜而難以去分析。因此，提供一個有效率且自動化的驗證技術是相當重要的。

軟體驗證所要分析的特性主要分成兩大類： **安全性質（Safety Property）**、**活化性質（Liveness Property）[11]**。安全性質，簡單的說：程式進行時不會進入不良的狀態。舉例來說，並行互斥（mutual exclusion）指的是一個系統資源不會同時被兩個以上的行程（process）進行存取，其所代表的就是一種安全性質。而活化性質表示為：程式執行時終究會進入某個良好的狀態。像是某個行程在系統中要求某種服務，而系統最終將提供此服務給請求的行程，如此將不會產生飢餓的情形（freedom from starvation），此即為一種活化性質。

在本篇論文中所討論的活化性質將以 **progress property** 為主。所謂的 **progress property** 所聲稱的是，無論如何描述系統，某些特定的行動**(specified action)**最終**(*eventually*)**將會被執行。明顯的 ***progress*** 的相反面即為 ***starvation***。**progress property** 能簡單並且足夠詳細的描述並行系統中的活化性質。

模型驗證工具目前大多以全域分析為主要架構。但模型驗證目前碰到的最主要問題為組態爆炸，這樣的結果使的許多以全域分析的驗證工具在還沒有能檢驗出系統是否違反某些性質時，就因為組態爆炸的產生而導致驗證失敗。近年來對此問題提出相當多的方法來減緩組態爆炸，而眾所皆知比較具有潛力的解決方案為使用**局部性分析（Compositional Reachability Analysis）[17][19]**來進行模型驗證，局部性分析主要原理是利用*divide and conquer* 的技術，讓驗證工具先去檢驗子系統，最後再把子系統分析的結果結合起來，藉以達到分析原有系統的目的。

局部性分析能夠成功的基礎在於軟體系統可建構成好的階層式架構，但並非所有的軟體系統都能符合這樣的情況。對此Cheng[33][34]提出一種方法稱模型架構重構（Model architecture Refactoring）。利用此技術可以在不改變原始系統行為的情況下，將系統模型架構轉換成可分析的階層架構。如此就可順利利用局部性分析來檢驗軟體性質。不過，局部性分析依然有他的極限存在。欲分析的軟體性質必須保留到最後的全域階層而無法在子階層中做驗證。如果欲檢驗的性質過多將造成子系統的同步溝通行為必須保留，無法將其化減為內部行為來進行最小化動作，如此大幅消減局部性分析的效果。對於此問題Cheung在CSP based的模型系統中對於檢驗安全性質[28][29]和活化性質[30]各提出一種技巧來加強局部性分析的效能。在檢測安全性質時，利用在適當的合成階層中加入一種特別的狀態 π，來代表系統中出現違反的狀態；而在檢測活化性質時，則利用一種全域的特殊行動（action），將其加入到子系統中，最後於全域系統中檢查狀態中是否仍存在此全域特殊行動。兩種方式各有巧妙並更加發揮局部性分析的優點。綜合局部性分析的優點，如果能將上述所有局部性分析的關鍵性技術融合於一體，將使軟體驗證工具具有更好的實用性與效能。

目前的研究顯示模型架構重構在 CCS based 的模型系統中產生較好的效果，如果使用 CSP based 的模型來進行重構，將無法真正發揮模型架構重構的效用。

因此我們以 Cheung 在檢驗系統性質的方法為基礎，透過許多改良與修正，使其能夠在 CCS base 的系統架構中檢驗系統安全性質和活化性質。

ArCats是由Cheng[35]所率領開發的局部性分析工具。其本身即具有模型架構重構的功能，這解決了局部性分析最重要的關鍵問題。ArCats是使用gcc語言來實做，這使得在跨平台的移植更加容易；更重要的是ArCats提供相當多自動化和人性化的地方，讓困難容易出錯的工作交由系統來處理。例如，在決定子系統中哪些行動必須和其他子系統做同步溝通時，如果交由人手來做輸入的動作，將有機會輸入錯誤，更何況決定哪些行動並需和其他子系統做溝通並不是一件容易的事情，此時可以通過ArCats的功能來幫助我們解決這複雜的問題。然而ArCats並非毫無缺點，對於檢驗系統性質目前只支援到檢驗系統中某些簡單的特性，如是否存在死結（deadlock）；而對於檢驗其他的安全性質或活化性質並未支援。事實上以目前局部性軟體檢驗工具中，支援模型架構重構的功能是相當稀少，如果我們將檢驗工具架構在ArCats上將大幅增強模型驗證的效能，也縮短了整個系統的開發時間。本篇論文將著重利用ArCats tool的各種優點，配合之前所述的檢驗系統性質的各種技巧，開發出可檢驗系統安全性質和活化性質的工具。

1.2 論文架構

本篇論文的結構如下：在第二章，首先我們將先回顧一些必須的背景資料，像是 Labeled transition system（LTS）、Transitional Semantic of LTS、parallel composition（平行合成）、Compositional Reachability Analysis（局部性分析）、狀態空間（state space）列舉的方式和幾種狀態空間減化的方法，第二章最後將介紹 ArCats，ArCats 為一種局部性模型驗證工具，利用其特性和目前其他驗證工具作比較。在第三章中，我們將介紹如何將檢驗系統安全性質實作於 ArCats 裡，

並透過特殊的方法來增強在局部性分析中的檢驗效果。第四章中，同樣的我們將介紹如何在子系統中加入特殊的全域行動來檢驗活化性質，並將檢驗活化性質的功能實做於 ArCats 裡。第五章中，我們將分析比較使用一般局部性模型分析和利用 ArCats 來進行局部性模型分析的差異，並透過 ArCats 所提供最小化功能，來顯示系統在檢驗系統性質時的效能與成果。最後第六章我們將說明經由我們在 ArCats 中所加入的功能，如何將 ArCats 提升到另一個層次，並述說未來可能的繼續的研究內容。