Τμήμα Πληροφορικής Α.Π.Θ.

Μεταπτυχιακό Πρόγραμμα Σπουδών

Ακαδημαϊκό Έτος 2006-2007

Τυπικές Μέθοδοι Ανάλυσης Συστημάτων: Ο Ελεγκτής Μοντέλων SPIN

Τρίτη 24 Απριλίου 2007

Ιστορική αναδρομή: Το κόστος της αποτυχίας

- Το Ariane-5 εκτοξεύθηκε στις 4 Ιουνίου 1996, για να συντριφθεί 39 δευτερόλεπτα αργότερα λόγω ενός σφάλματος στο λογισμικό ελέγχου.

  (Ο Ariane-5 αυτοκαταστράφηκε εξαιτίας μιας προσπάθειας του συστήματος πλοήγησης να μετατρέψει έναν 64-bit αριθμό σε 12-bit δημιουργώντας λάθος υπερχείλισης)

  Κόστος κατασκευής : Πάνω από 7 δισεκατομμύρια $

  Χρόνος: 10 έτη
- Το 1994 ανακαλύφθηκε ένα λάθος (bug) στο floating point hardware του μικροεπεξεργαστή Pentium της Intel.Το κόστος αντικατάστασης/ διόρθωσης του ελαττώματος κυμάνθηκε πάνω από 400 εκατομμύρια $.
- Η Intel σήμερα διαθέτει μια από τις μεγαλύτερες ομάδες που ασχολούνται αποκλειστικά με την ανάλυση συστημάτων με την βοήθεια των τυπικών μεθόδων

Τυπικές Μέθοδοι Ανάλυσης

- Ανάγκη για ανάλυση και επαλήθευση συστημάτων:
  - Συστήματα πληροφοριών αεροναυτικής και αεροπλοήγησης
  - Κρίσιμα συστήματα ασφαλείας
  - Ενθυλακωμένα Συστήματα (embedded systems)
  - E-banking
  - E-shopping
  - E-health
  - …

Ο ελεγκτής μοντέλων SPIN (The SPIN model checker) (1/ 6)

Έλεγχος Μοντέλων (Model Checking):
Επαναστατική μέθοδος για την αυτοματοποιημένη επαλήθευση συστημάτων. Ελέγχει κατά πόσο ένα μοντέλο (αντιπροσωπευτικό του συστήματος που ελέγχουμε) ικανοποιεί μια ιδιότητα π.χ.

- **Το σύστημα δίνει το σωστό αποτέλεσμα;**
- **Ικανοποιεί βασικές λειτουργικές ιδιότητες;**
- **Το σύστημα περιέχει αδιέξοδα (deadlock);**
- **Υπάρχει κώδικας που δεν εκτελείται (unreachable code);**
- **…**

Ο ελεγκτής μοντέλων SPIN αναπτύχθηκε στα Bell Labs από τον G. Holzmann στο Unix group of the Computing Sciences Research Center, το 1991

Ο ελεγκτής μοντέλων SPIN (The SPIN model checker) (2/ 6)

Ο έλεγχος μοντέλων απαιτεί μια ακριβή και ξεκάθαρη δήλωση των ιδιοτήτων που θα ελεγχθούν. Αυτό γίνεται συνήθως σε μια χρονική λογική (temporal logic).

- Το SPIN χρησιμοποιεί την γραμμική χρονική λογική (LTL, Linear Temporal Logic)

[]	(operator always)
<>	(operator eventually)
!	(operator for negation)
U	(operator strong until)
V (the dual of U): (p V q) means !(!p U !q))	
&&	(operator for logical and)
||	(operator for logical or)
/ \	(alternative form of &&)
\/	(alternative form of ||)
->	(operator for logical implication)
<->	(operator for logical equivalence)

Π.χ. []!P

(´always not P´) :

where P=-3 in an Elev model

P_{Elev}={-2,-1,0,1,2,3,4,5,6,7,8}

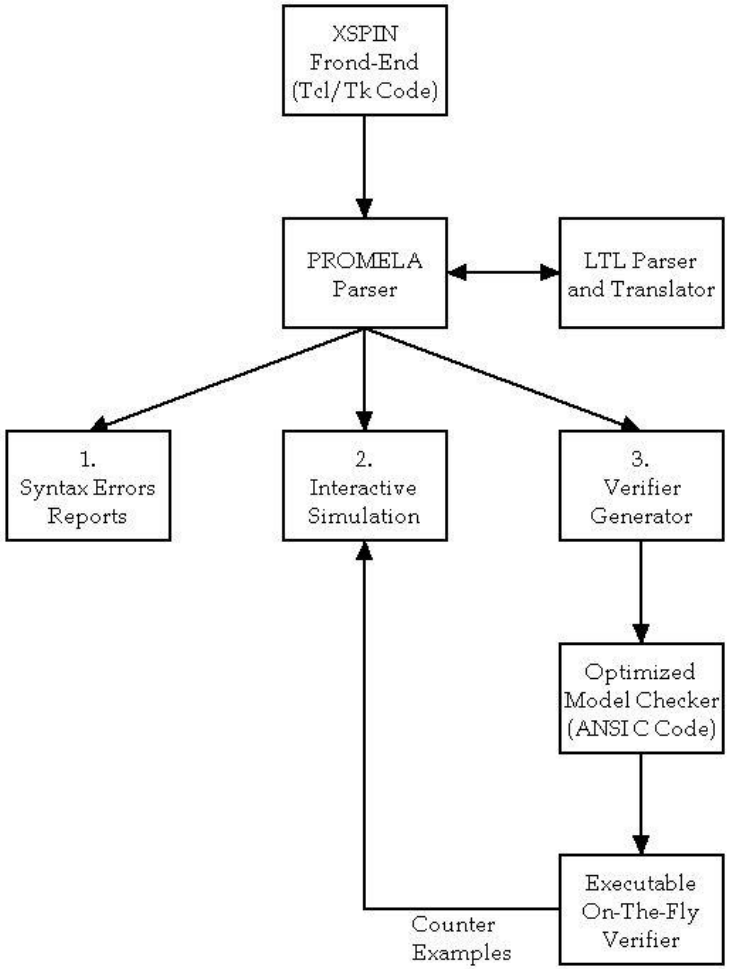
Ο ελεγκτής μοντέλων SPIN (The SPIN model checker) (3/ 6)

- Πρόκειται για έναν on-the-fly ελεγκτή μοντέλων
  (Δεν αποθηκεύεται ολόκληρος ο γράφος των καταστάσεων ενός συστήματος και ειδικά οι μεταβάσεις ελέγχου, αλλά μόνο οι τελικές καταστάσεις του)
- Το SPIN πραγματοποιεί έναν συστηματικό έλεγχο του συνόλου καταστάσεων ενός συστήματος (με χρήση διαφορετικών αλγόριθμων αναζήτησης όπως depth-first και breadth-first)
- Χρησιμοποιεί διάφορες τεχνικές μείωσης του συνόλου καταστάσεων ενός συστήματος προς αποφυγή του φαινομένου της ‘έκρηξης του χώρου καταστάσεων΄ (State-Space Explosion problem)
  - Partial Order Reduction
  - Abstraction
  - Συμμετρικές Τεχνικές
  - Τεχνικές Συμπίεσης

Ο ελεγκτής μοντέλων SPIN (The SPIN model checker) (4/ 6)

- Το SPIN παρέχει έναν προσομοιωτή ο οποίος και δίνει την δυνατότητα στους αναλυτές να αποκομίσουν χρήσιμα αποτελέσματα μέσω της 'εκτέλεσης' του μοντέλου τους (κατανόηση του εν παραγωγή χώρου καταστάσεων, έλεγχος αποτελεσματικότητας/ περατότητας μεταβάσεων του συστήματος).
  - Interactive
  - Random
  - Guided
- Αποτελεί έναν 'εξαντλητικό αναλυτή' του χώρου καταστάσεων επαληθεύοντας τις ιδιότητες που πρέπει να ικανοποιεί το προς εξέταση μοντέλο
- Εύκολη χρήση του SPIN μέσου του γραφικού περιβάλλοντος XSPIN (tcl/ tk)
- Υλοποίηση/ Κωδικοποίηση των μοντέλων σε PROMELA

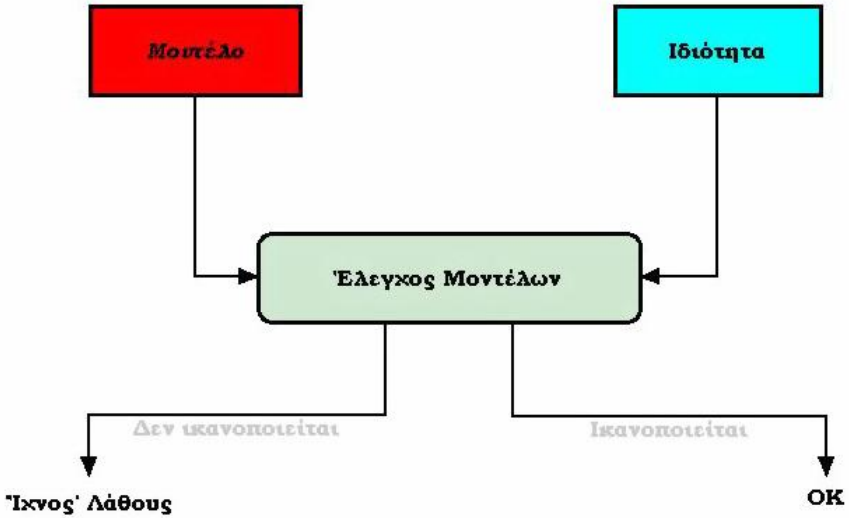
Ο ελεγκτής μοντέλων SPIN (The SPIN model checker) (5/ 6)



Η δομή της interactive προσομοίωσης και επαλήθευσης του εργαλείου SPIN

Ο ελεγκτής μοντέλων SPIN (The SPIN model checker) (6/ 6)

Μετά τη διόρθωση συντακτικών λαθών, αν βέβαια υπάρχουν, το σύστημα εισάγεται σε interactive προσομοίωση, μέχρι να βεβαιωθούμε ότι συμπεριφέρεται όπως θέλουμε και ορίζεται. Έπειτα, το SPIN χρησιμοποιείται για να παράγει μία on-the-fly επαλήθευση του προγράμματος. Η επαλήθευση εκτελείται με διαφορετικές επιλογές κάθε φόρα και αν οδηγήσει σε σφάλμα, τότε γυρίζουμε ξανά στο στάδιο της interactive προσομοίωσης για να δούμε τι προκάλεσε το σφάλμα. Αν είναι εφικτό το διορθώνουμε και κάνουμε την επαλήθευση ξανά.



PROMELA (PROcess MEta LAnguage)

1/ 5

- **Πρόκειται για μια γλώσσα αναπαράστασης και μοντελοποίησης**
- **C-like σύνταξη**
- **Αναπαράσταση επικοινωνίας με χρήση σύγχρονων/ ασύγχρονων καναλιών ανταλλαγής μηνυμάτων**
- **Οι προτάσεις (statements) στην PROMELA είναι είτε εκτελέσιμες (executable) είτε μπλοκαρισμένες (blocked)**
- **Χρησιμοποιείται κυρίως για την αφηρημένη μοντελοποίηση πρωτοκόλλων επικοινωνίας [HOLZ97]**
- **Ταιριάζει απόλυτα επίσης στην μοντελοποίηση πρακτόρων επικοινωνίας. Η «συζητήσεις» μεταξύ πρακτόρων αναπαρίστανται σαν διεργασίες, τα μονοπάτια αυτών των συζητήσεων μοντελοποιούνται σαν κανάλια και οι μεταβλητές που χρησιμοποιούνται ορίζονται και ελέγχονται χωρίς καμία αλλαγή.**

PROMELA (PROcess MEta LAnguage)

2/ 5

- Έτσι, προτάσεις μπορεί να μπλοκαριστούν από μία εντολή if εάν η συνθήκη αυτής είναι false. Οι προτάσεις αυτές μπορούν να εκτελεστούν την στιγμή εκείνη που η συνθήκη θα γίνει true
- Κάτι τέτοιο παρέχει μηχανισμούς συγχρονισμού επικοινωνιών μεταξύ διεργασιών έχοντας ως αποτέλεσμα μια εκτελέσιμη διεργασία (responder) να περιμένει για ένα μήνυμα από μια άλλη διεργασία (initiator).
- Είναι μη-ντετερμινιστική γλώσσα η οποία χρησιμοποιεί τρεις βασικούς τύπους αντικειμένων
  - Processes (global)
  - Channels (local/ global)
  - Variables (local/ global)
- Π.χ.

```
byte count, total = 0;
channel com_a_b = [2] of {byte};
a == b;
run test(); //proctype test promela model
```

PROMELA (PROcess MEta LAnguage)

3/ 5

- **Πίνακες**
  - **Π.χ.** `int table[max]`
- **Συμβολικές απαριθμημένες σταθερές**
  - `mtype = {LINE_CLEAR, TRAIN_ON_LINE, LINE_BLOCKED}`
- **Δομές**
  - `typdef msg {byte data[4], byte checksum}`
- **Σταθερές**
  - `#define MAX 999`
- **Τελεστές**
  - `+, -, *, /, %, --, ++, //αριθμητικοι`
  - `>, >=, <, <=, ==, !=, //συσχετισης`
  - `&&, ||, !, //λογικοι`
  - `!, ?, //καναλιων επικονωνιας`

PROMELA (PROcess MEta LAnguage)

4/ 5

- **Διεργασία (process)**
  - ```
    proctype ProcessA()
    {
    byte newVariable;
    newVariable = 3
    }
    ```
- **Η Διεργασία init (παρόμοια λειτουργία με την main στην C)**
  - ```
    init
    {
    run ProcessA();
    run ProcessB()
    }
    ```

Σε αυτήν την περίπτωση, η λέξη κλειδί run πυροδοτεί τις δύο διεργασίες.

PROMELA (PROcess MEta LAnguage)

5/ 5

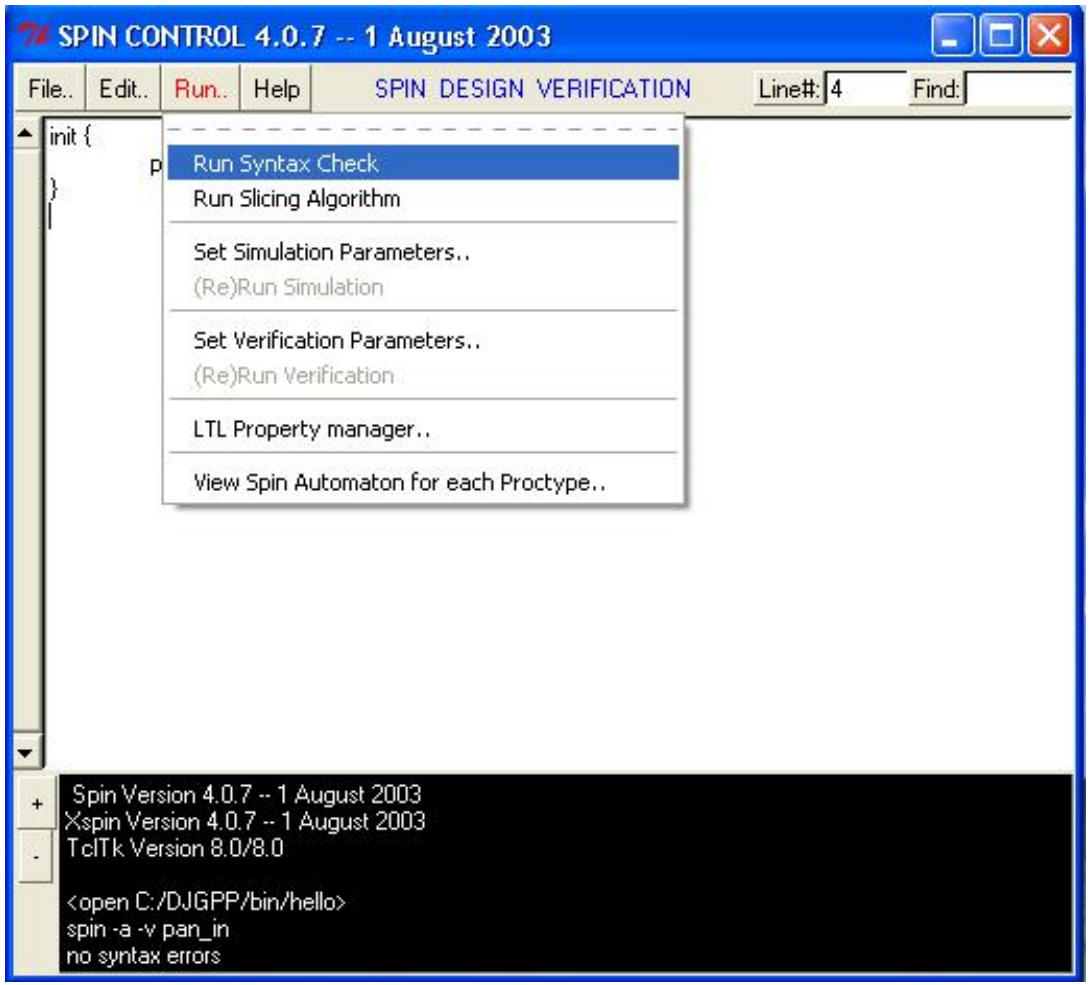
- **Κανάλια (Channels)**
  - `channel com_a = [2] of {byte};`
- **Αποστολή μηνύματος (! operator)**
  - `com_a!4`

  →Η προηγούμενη πρόταση μπλοκάρεται εκτός εάν στο κανάλι com_a υπάρχει μια κενή θέση)
- **Λήψη μηνύματος (? operator)**
  - `com_a?msg`

  →(Η προηγούμενη πρόταση μπλοκάρεται εκτός εάν στο κανάλι com_a περιέχει μια οποιαδήποτε τιμή τύπου byte)

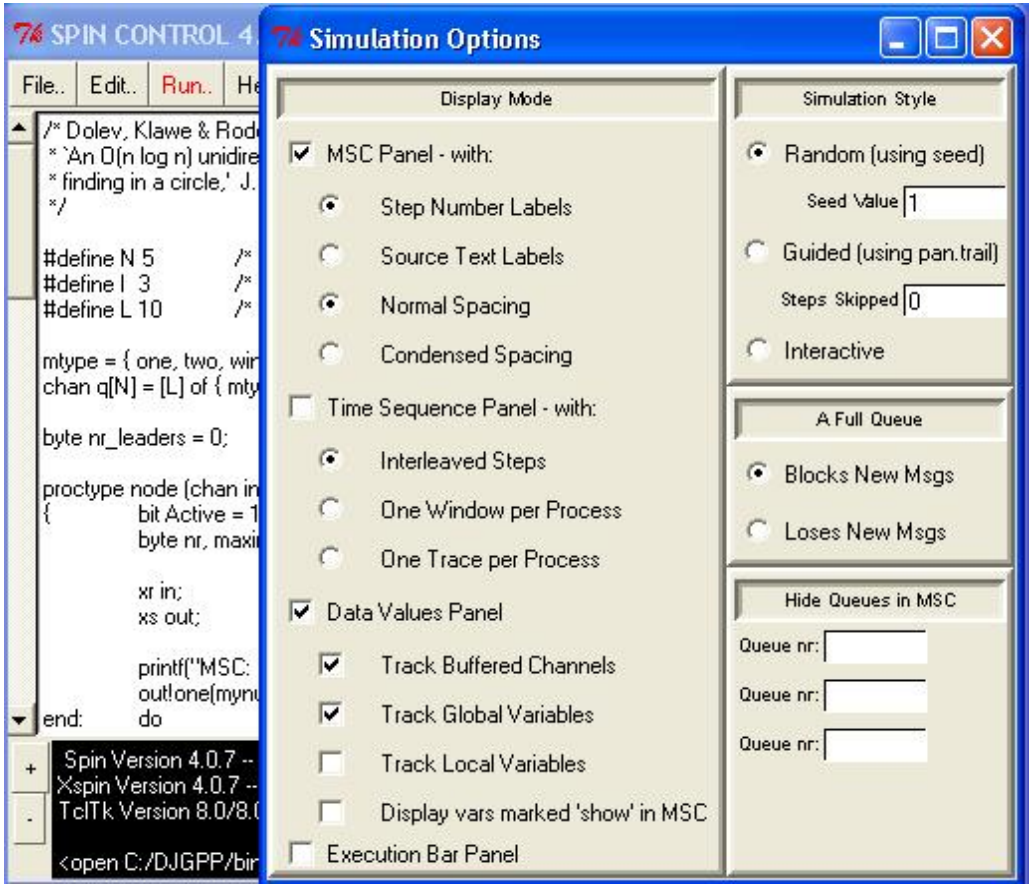
XSPIN: Παράδειγμα (1/ 7)

Μοντελοποιούμε το σύστημα που θέλουμε να επαληθεύσουμε και εκτελούμε συντακτικό έλεγχο (syntax check):



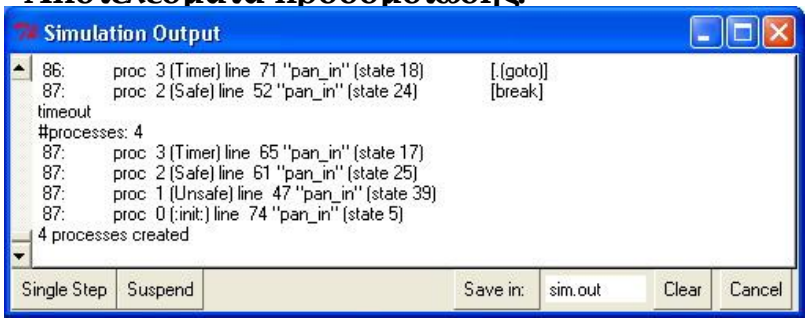
XSPIN: Παράδειγμα (2/ 7)

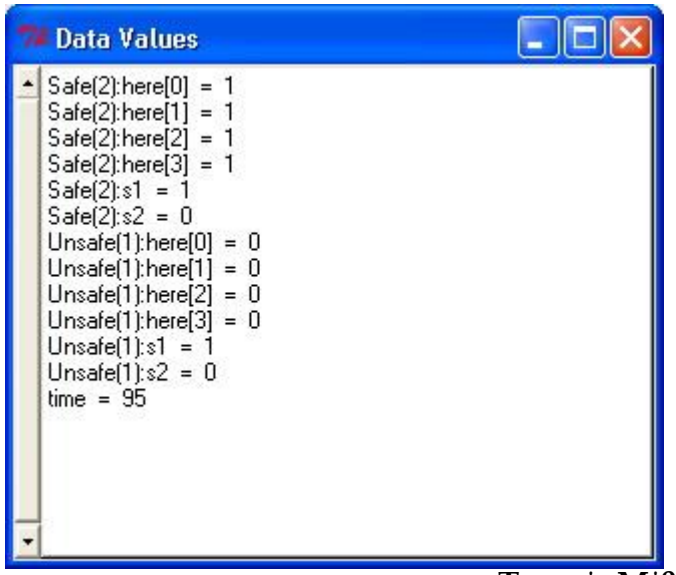
Θέτουμε τις παραμέτρους προσομοίωσης ανάλογα με τα αποτελέσματα που επιθυμούμε:

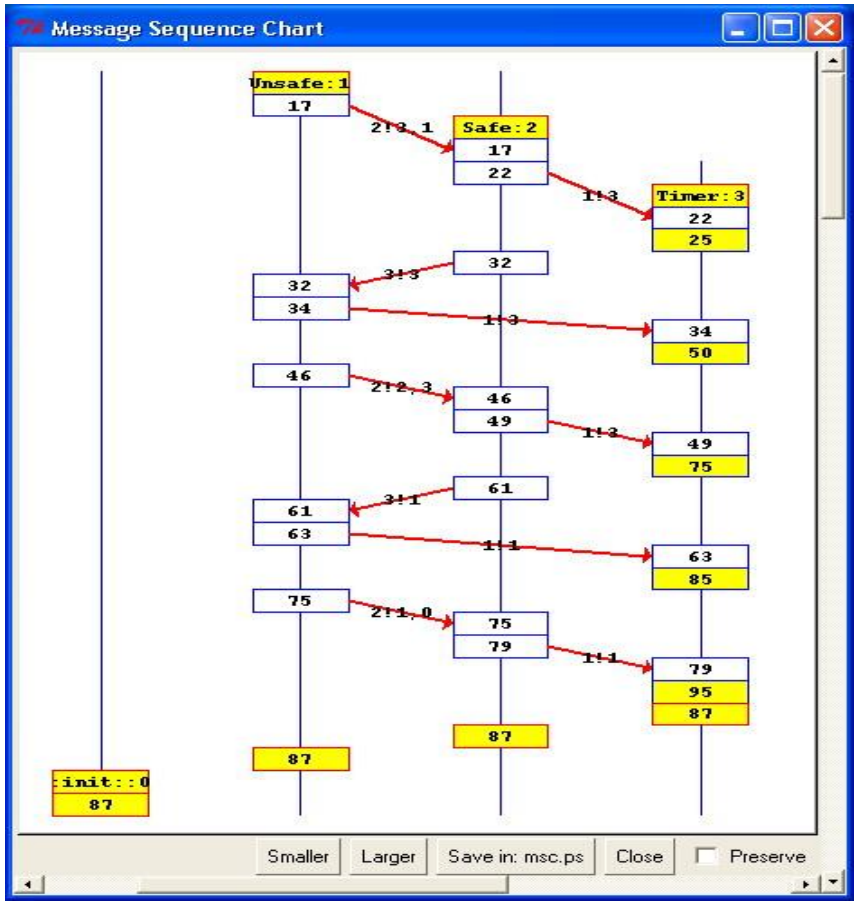


XSPIN: Παράδειγμα (3/ 7)

Αποτελέσματα προσομοίωσης:

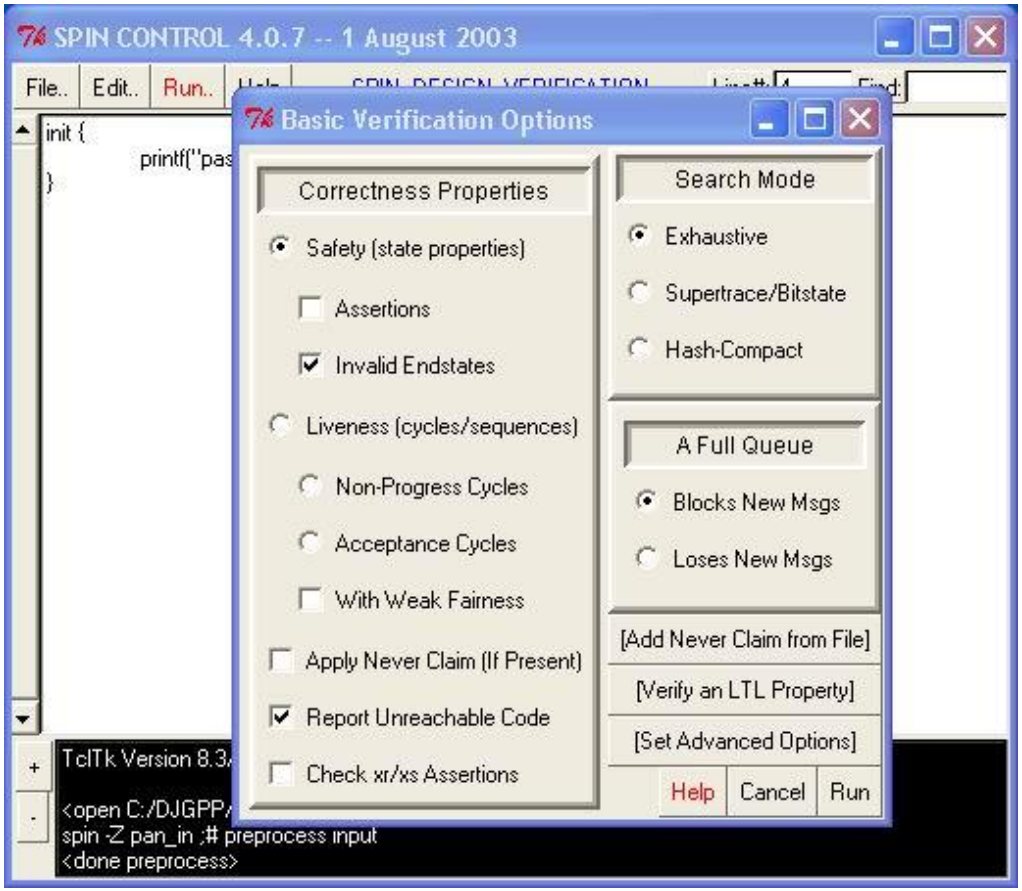






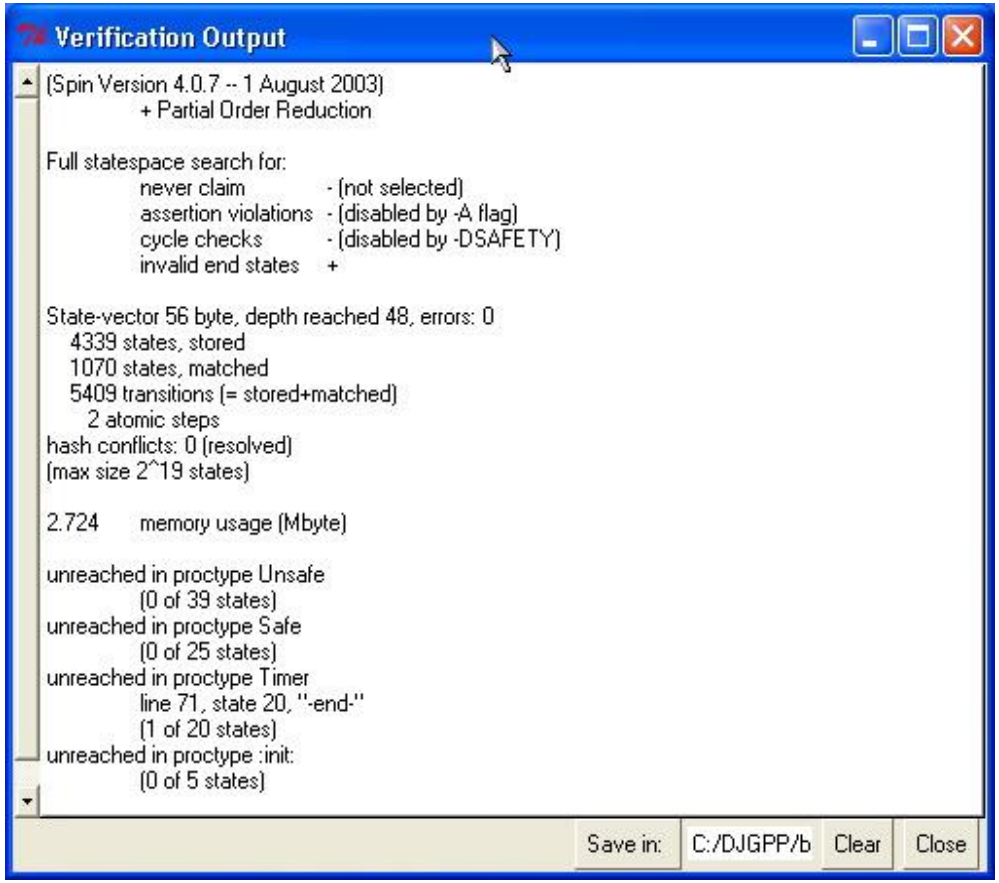
XSPIN: Παράδειγμα (4/ 7)

Επιλέγουμε τις παραμέτρους για την επαλήθευση του μοντέλου (πχ invalid endstate, unreachable Code καθώς και τον τύπο αναζήτησης του χώρου των καταστάσεων που θα παραχθεί) :



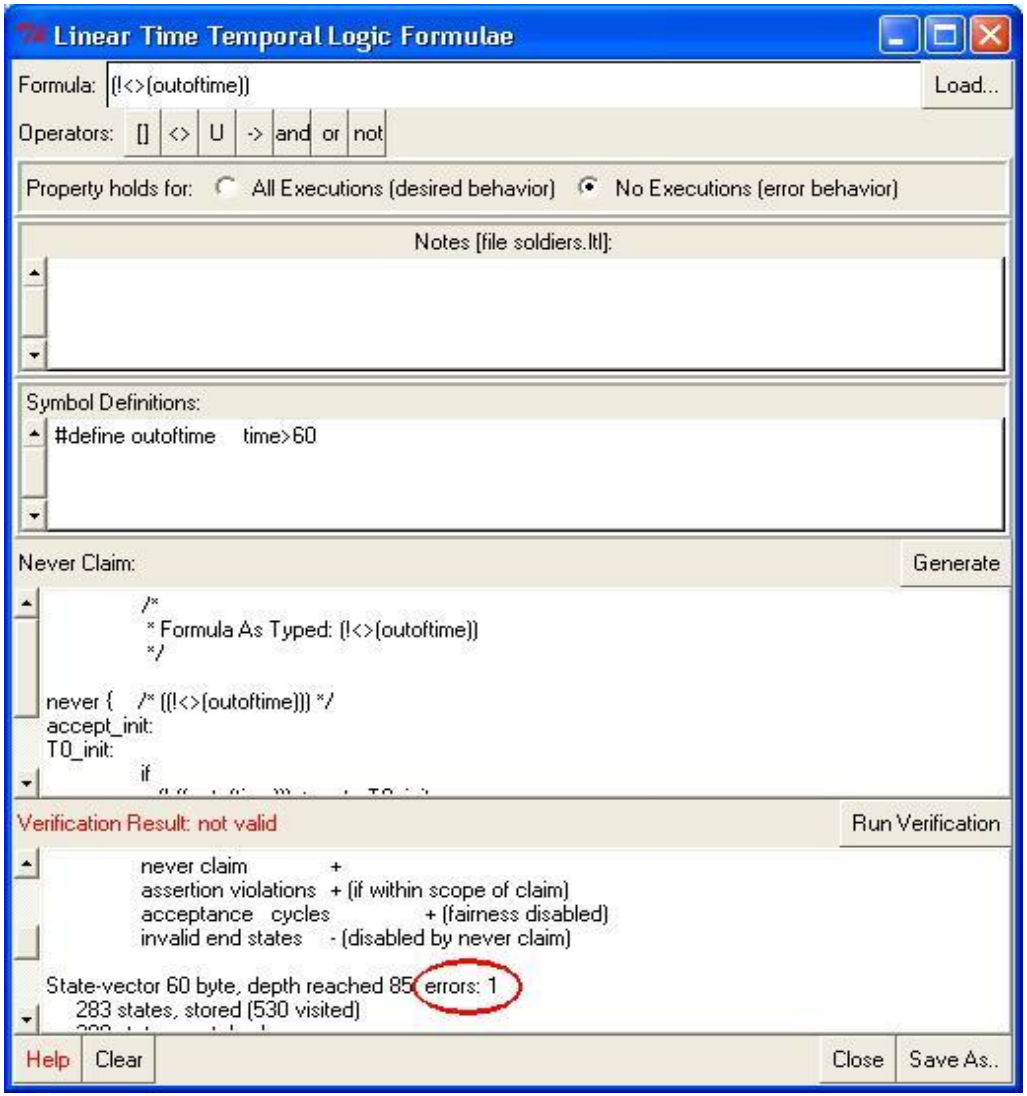
XSPIN: Παράδειγμα (5/ 7)

Αποτελέσματα επαλήθευσης (`errors:0`):



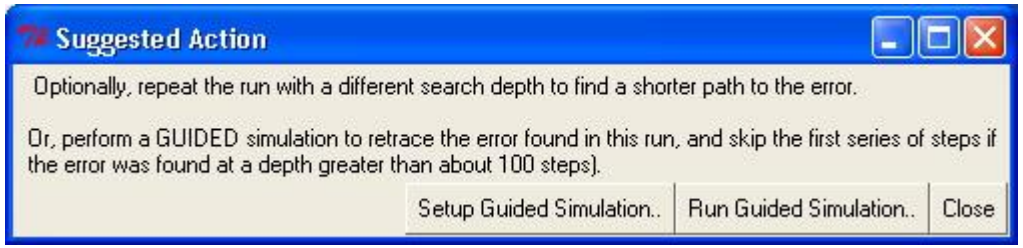
XSPIN: Παράδειγμα (6/ 7)

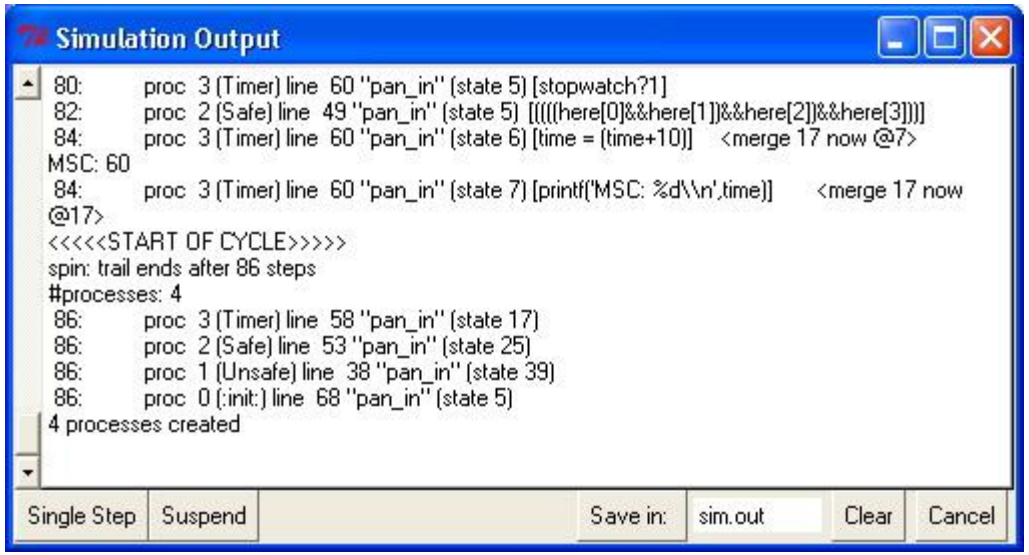
LTL Manager

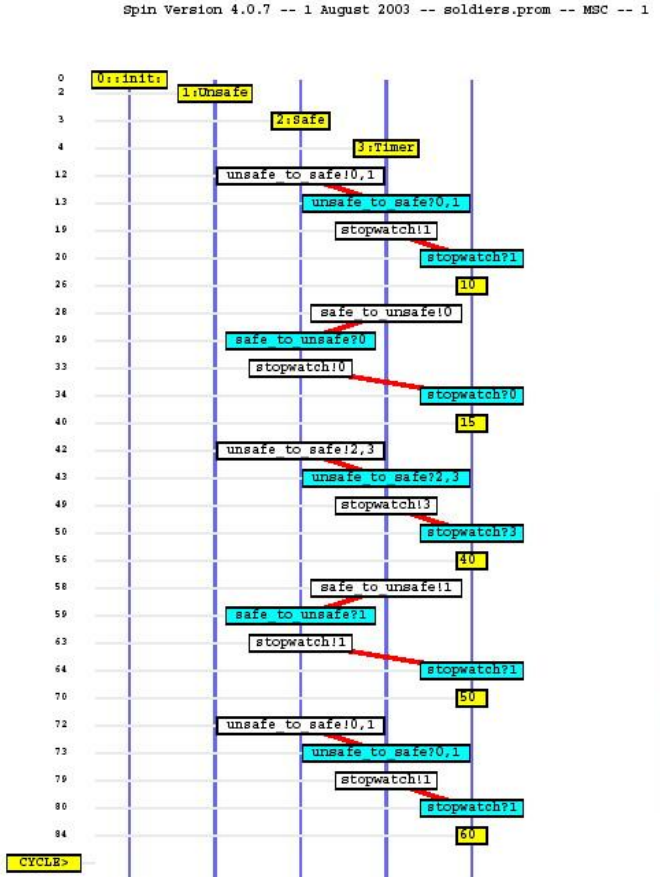


XSPIN: Παράδειγμα (7/ 7)

Προσομοίωση Guided για να βρεθεί το λάθος (με την βοήθεια του counter-example)







Εφαρμογές του SPIN (1/ 2)

- Χρήση του SPIN για την επαλήθευση κατανεμημένων συστημάτων πληροφοριών
- Κρίσιμα συστήματα ασφαλείας
- Μοντελοποίηση και επαλήθευση πρωτοκόλλων επικοινωνίας
- ...

Μερικά παραδείγματα:

- Επαλήθευση του συστήματος πληροφοριών για τον έλεγχο λειτουργικότητας των αλγορίθμων στο φράγμα του Λογισμικό επεξεργασίας κλήσεων σε τηλεφωνικά κέντρα μεταφοράς δεδομένων (Pathstar designed by Lucent Tech.
- Σε επιλεγμένους αλγόριθμους σε έναν μεγάλο αριθμό εφαρμογών διαστημικών αποστολών όπως Deep Space 1, Cassini,Mars Exploration Rovers, Deep Impact (NASA Formal Method Group

Εφαρμογές του SPIN (2/ 2)

- Interlocking control by Distributed Signal Boxes: Design and Verification with the SPIN model checker (see references)
- Πρόκειται για την ανάπτυξη ενός κατανεμημένου αλγορίθμου (DSBs) με την βοήθεια του SPIN
- Επιτυχής εφαρμογή του σε ένα συγκοινωνιακό δίκτυο όπου και γίνεται ο έλεγχος πρόσβασης σε κρίσιμα σημεία (Συγκοινωνιακό δίκτυο του ΜΕΤΡΟ της Αθήνας)
- Intrusion attack tactics for the model checking of e-commerce security guarantees
- Σχεδιασμός και μοντελοποίηση ενός intruder για την εύρεση σφαλμάτων σε πρωτόκολλα επικοινωνίας
- Δημιουργία ενός intruder framework όπου ο και εφαρμόζονται πολλαπλές επιθέσεις στο ύπο εξέταση πρωτόκολλο

- Deadlock freedom
- no assertion violation
- no invalid end-states

Επίλογος

- Επαλήθευση ιδιοτήτων ανάλογα με τις συγκεκριμένες λειτουργίες και καταστάσεις που θέλουμε το μοντέλο να επαληθεύει
  - Επαλήθευση των επιβεβαιώσεων που θέλουμε να ισχύουν (assertion violation checking)
  - Επαλήθευση ιδιοτήτων επιθυμητών ή μη/ επιθυμητών καταστάσεων τερματισμού
  - Εποπτεία, επανασχεδιασμός και διόρθωση του μοντέλου με την βοήθεια του προσομοιωτή του SPIN
- Μοντελοποίηση πολύπλοκων συστημάτων μειώνοντας δραστικά των παραγόμενο χώρο καταστάσεων μέσω διαφόρων τεχνικών
- Εύκολη μοντελοποίηση μέσω της PROMELA
- Εύχρηστο περιβάλλον μοντελοποίησης μέσω του XSPIN

Recommended Books/ References/ Sources (1/ 2)

- G. J. Holzmann. Design and Validation of Computer Protocols.Prentice-Hall,Englewood Cli.s, New Jersey, USA, 1991.
- G. J. Holzmann. The model checker SPIN. IEEE Transactions on Software Engineering,1997
- G.J. Holzmann, "Design and Validation of Protocols: A Tutorial," Computer Networks and ISDN Systems, vol. 25, no. 9, pp. 981-1,017, 1993.
- Holzmann, G.J. "The Spin Model Checker - Primer and Reference Manual", Addison-Wesley Publ., ISBN 0-321-22862-6, available September 2003.
- Official SPIN website http:// spinroot.com/ spin/ whatispin.html
- The World Wide Web Formal Methods Virtual Library http:// www.afm.sbu.ac.uk/
- The Basic SPIN manual (see SPIN´s official website)
- Concise PROMELA Reference (see SPIN's official website)

Recommended Books/ References/ Sources (2/ 2)

- Carnegie Mellon University – Formal methods Group
  http:// www.cs.cmu.edu/ groups/ formal-methods/ formal-methods.html
- NASA formal methods website
  http:// eis.jpl.nasa.gov/ quality/ Formal_Methods/
- Formal Methods & Tools (FMT) group at the University of Twente, The Netherlands
  http:// fmt.cs.utwente.nl/
- Basagiannis, S., Katsaros, P. and Pombortsis, A., "Interlocking control by Distributed Signal Boxes: design and verification with the SPIN model checker", Lectures Notes in Computer Science 4330, 317-328, Springer Verlag, Sorrento, Italy 4-6 Dec.for The Fourth International Symposium on Parallel and Distributed Processing and Applications, ISPA'2006.
- Basagiannis, S. Katsaros, P. and Pombortsis, A.,"Intrusion attack tactics for the model checking of e-commerce security guarantees" (submitted)

Επικοινωνία

Επικοινωνία:

Παναγιώτης Κατσαρός
katsaros@csd.auth.gr

Στυλιανός Μπασαγιάννης
basags@csd.auth.gr