ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

ArrayTracer: Ανάλυση και Οπτικοποίηση της Επίδοσης Παράλληλων Εφαρμογών.

Ζάρρας Απόστολος

Μεταπτυχιακή Εργασία

Ηράκλειο, Ιούλιος 1997

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

ArrayTracer: Ανάλυση και Οπτικοποίηση της Επίδοσης Παράλληλων Εφαρμογών.

Εργασία που υποβλήθηκε από τον
Ζάρρα Απόστολο
ως μερική εκπλήρωση των απαιτήσεων
για την απόκτηση
ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΕΙΔΙΚΕΥΣΗΣ

Συγγραφέας:

Ζάρρας Απόστολος
Τμήμα Επιστήμης Υπολογιστών

Εισηγητική Επιτροπή:

Νικολάου Χρήστος, αναπληρωτής καθηγητής, Επόπτης

Χούστη Αικατερίνη, αναπληρώτρια καθηγήτρια, Μέλος

Μαρκάτος Ευάγγελος, επίκουρος καθηγητής, Μέλος

Δεκτή:

Πάνος Κωνσταντόπουλος, αναπληρωτής καθηγητής
Πρόεδρος Επιτροπής Μεταπτυχιακών Σπουδών

Ηράκλειο, Ιούλιος 1997

Στην κα. Βούλα στον κ. Βασίλη
και στην καλή νεράιδα...

ArrayTracer: Ανάλυση και Οπτικοποίηση της Επίδοσης Παράλληλων Εφαρμογών

Ζάρρας Απόστολος
Μεταπτυχιακή Εργασία

Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

Περίληψη

Ο *ArrayTracer* είναι ένα εργαλείο ανάλυσης και οπτικοποίησης της επίδοσης μιας παράλληλης εφαρμογής. Η σχεδίαση και υλοποίησή του ξεκίνησε πριν από 4 χρόνια από τον Τίτο Σαρειδάκη [**?**]. Σταδιακά βελτιώθηκε και ολοκληρώθηκε. Οδηγήθηκε σε μια μορφή που ικανοποιεί τις σύγχρονες απαιτήσεις των χρηστών. Κυριότερες από αυτές είναι η ευκολία στη χρήση, η ελάχιστη επίδραση στη ροή ελέγχου της εφαρμογής, η ελαχιστοποίηση του χώρου αποθήκευσης των ιχνών, η γρήγορη επεξεργασία τους και η επαρκής απεικόνιση. Τα βασικά δομικά στοιχεία του εργαλείου είναι:

- Τμήμα Στατικής Ανάλυσης.
- Τμήμα Ιχνοληψίας.
- Τμήμα Ανάλυσης των Ιχνών.
- Τμήμα Οπτικοποίησης των Ιχνών.

Τα δύο πρώτα ανήκουν στην πρώτη φάση της υλοποίησης και τα υπόλοιπα δύο είναι το αντικείμενο της παρούσας εργασίας.

Η διαδικασία ανάλυσης γίνεται μετά το τέλος της εκτέλεσης της εφαρμογής *(Post Mortem)*. Προσπαθώντας να ελαττώσουμε την καθυστέρηση στον χρόνο εκτέλεσης, που οφείλεται στην εισαγωγή του κώδικα καθοδήγησης χρησιμοποιούμε ίχνη μειωμένου μεγέθους. Δεν γίνεται εξαγωγή χρονοσφραγίδας και για κάθε γεγονός κατανάλωσης χρόνου εξάγεται μόνο ένα ίχνος.

Η ανάλυση έχει ως στόχο την παραγωγή της ακολουθίας των γεγονότων για την κάθε διεργασία καθώς και των συγκερασμό τους σε μία ακολουθία. Η διαδικασία αυτή στηρίζεται σε εξαρτήσεις μεταξύ των γεγονότων και έχει ως αποτέλεσμα την παραγωγή λογικών χρονοσφραγίδων. Οι εξαρτήσεις προκύπτουν από τη σειριακό–τητα του κώδικα, την ενδοεπικοινωνία και τον συγχρονισμό μεταξύ των διεργασιών της εφαρμογής.

Η απεικόνιση της ακολουθίας αυτής γίνεται από μια επέκταση του *Paragraph*. Η διασύνδεση με το *Paragraph* έγινε με χρήση ενός φίλτρου που μετατρέπει τα ίχνη που παράγει ο *ArrayTracer* σε γεγονότα απεικόνισης μορφής *PICL* [**?**]. Το *Paragraph* επεκτάθηκε με την προσθήκη νέων διαγραμμάτων, ώστε να καλύπτει τις δυνατότητες του *ArrayTracer*.

ArrayTracer: Parallel Performance Analysis and Visualization

Zarras Apostolos
Master's Thesis

Computer science Department
University of Crete

Abstract

ArrayTracer is a parallel performance analysis and visualization tool. The design and implementation began 4 years ago, by Titos Saridakis [**?**]. During these years it's been developed and enhanced according to the modern ideas in the field of performance analysis tools.

ArrayTracer's basic components are the following :

- Static Analysis Module.
- Trace Collection Module.
- Trace Analysis Module.
- Visualization Module

The first two parts belong to the first phase of the tools implementation. The last two are the subject of this thesis.

The analysis procedure takes place at Post−Mortem time. We tried to reduce the dilation introduced by *ArrayTracer* at the application's execution time, by producing reduced size trace records. These trace records do not contain timestamps. Besides that we reduced the number of trace records dumped. For each time consuming event we only produce one trace record instead of two.

During the trace analysis procedure we must produce each process's event sequence and merge them in a total event sequence. To do that we construct logical timestamps according to dependencies that hold for the application processes. These dependencies result from the serial code executed, the interprocess communication and synchronization.

To visualize the results we linked *ArrayTracer* with *Paragraph*. This was done with the implementation of a filter that transforms *ArrayTracer*'s traces to *PICL* [**?**] trace format events. We extended *Paragraph* with new displays that cover *ArrayTracer*'s extra trace capabilities (Variables References, Subroutine/Function Calls).

Περιεχόμενα

iii

v

Περιεχόμενα **vii**

Κατάλογος Πινάκων **ix**

Κατάλογος Σχημάτων **xi**

Ευχαριστίες **1**

1 Εισαγωγή **3**

2 Θεωρητική Προσέγγιση στη διαδικασία οπτικοποίησης **7**
2.1 Εισαγωγή 7
2.2 Είδη Οπτικοποίησης 7
2.3 Σκοπός της Οπτικοποίησης 8
2.4 Χρόνος Οπτικοποίησης 9
2.4.1 Οπτικοποίηση σε πραγματικό χρόνο *(Real Time)* 9
2.4.2 Οπτικοποίηση μετά το τέλος εκτέλεσης της εφαρμογής *(Post Mortem)* 9
2.5 Στόχοι και τεχνικές σχεδίασης 9
2.5.1 Ευκολία στην κατανόηση 10
2.5.2 Ευκολία στη χρήση 10
2.5.3 Φορητότητα 10
2.6 Βασικές αρχές και προτάσεις για τη σωστή σχεδίαση ενός εργαλείου ανάλυσης και οπτικοποίησης προγραμματιστικών εφαρμογών. 10
2.7 Είδη Γραφικών Αναπαραστάσεων 11

3 Σύνοψη Εργαλείων Ανάλυσης και Οπτικοποίησης **13**
3.1 Εισαγωγή 13
3.2 Υπάρχον Λογισμικό 13
3.2.1 *BALSA* I−II 13
3.2.2 *ZEUS* 14
3.2.3 *PASTIS* 15
3.2.4 *Paragraph* 15
3.2.5 *TRACEVIEW* 16
3.2.6 *MAP−SHMAP* 17

3.2.7 *TOPSYS* 18
3.2.8 *SMILI* 18
3.2.9 *Pablo* 19
3.2.10 *Paradyn* 20
3.2.11 Σύνοψη των υπάρχοντων εργαλείων ανάλυσης και οπτικοποίησης 20

4 Τμήμα Ανάλυσης των Ιχνών **27**
4.1 Εισαγωγή 27
4.2 Φόρμες Ιχνοληψίας 28
4.3 Λογικές Χρονοσφραγίδες 29
4.3.1 Εισαγωγή: Βασικοί στόχοι του *ArrayTracer*. 29
4.3.2 Σχέση Διάταξης Lamport 36
4.3.3 Χρήση σχέσεων αναδιάταξης για την δημιουργία λογικών χρονοσφραγίδων 37
4.4 Μηχανισμός ανίχνευσης ελλειπούς πληροφορίας επικοινωνίας . . . 39
4.5 Τμήμα Ανάγνωσης Ιχνών 40
4.6 Τμήμα Επικοινωνίας 43
4.6.1 Επίπεδο Αποστολής Μηνυμάτων 43
4.6.2 Επίπεδο Παραλαβής Μηνυμάτων 44
4.6.3 Επίπεδο Ομαδοποίησης Διεργασιών 45
4.7 Τμήμα Πηγαίου κώδικα 46
4.8 Τμήμα Μεταβλήτών 46
4.9 Φίλτρο Παραγωγής Γεγονότων σε μορφή *PICL* 47

5 Τμήμα Οπτικοποίησης των Ιχνών **51**
5.1 Εισαγωγή 51
5.2 Περιγραφή των απεικονίσεων του *Paragraph* 51
5.2.1 Απεικονίσεις του τρόπου Χρησιμοποίησης των Επεξεργαστών *(Utilization Displays)* 52
5.2.2 Απεικονίσεις της Ενδοεπικοινωνίας μεταξύ των Επεξεργαστών *(Communication Displays)* 55
5.2.3 Απεικονίσεις Υπολογιστικών Εργασιών της Εφαρμογής . . . 60
5.2.4 Απεικονίσεις που σχετίζονται με τον *ArrayTracer* 61

6 Παραδείγματα Χρήσης του *ArrayTracer* **67**
6.1 Χρήση του *ArrayTracer* για Μελέτη της Επίδοσης μιας Παράλληλης Εφαρμογής 67
6.1.1 Εισαγωγή 67
6.1.2 Περιγραφή της Εφαρμογής 67
6.1.3 Συντελεστές Απόδοσης 68
6.1.4 Μελέτη Αποτελεσμάτων 69
6.2 Χρήση του *ArrayTracer* για ανίχνευση προβλημάτων απόδοσης . . . 72
6.2.1 Εισαγωγή 72
6.2.2 Μελέτη της απόδοσης με χρήση του *ArrayTracer* 73
6.2.3 Βελτίωση της απόδοσης της εφαρμογής 73

7 Χρήση του *ArrayTracer* μέσω του Παγκόσμιου Ιστού **83**
7.1 Εισαγωγή 83
7.2 Σύστημα Διεπαφής με το χρήστη 84
7.2.1 Βασική Σελίδα Διεπαφής με τον Χρήστη 84
7.2.2 Επιμέρους Σελίδες Διεπαφής με το Χρήστη 85
7.3 Θέματα Ασφάλειας 88
7.4 Μελλοντικές κατευθύνσεις 90

Α Σύνολο Εντολών Εισόδου **93**

Β Διασύνδεση με το *Paragraph* **95**
Β.1 *PICL* γεγονότα που σχετίζονται με τον *ArrayTracer* 95
Β.2 Βιβλιοθήκη Απεικονίσεων 96

Γ Συντήρηση του *ArrayTracer* : Μετατροπές–Επεκτάσεις **99**
Γ.0.1 Προεπεξεργαστής 99
Γ.0.2 Ακρίβεια Μετρήσεων 100
Γ.0.3 Δημιουργία Ιχνών Ανεξάρτητων της Αρχιτεκτονικής του Κόμβου στον οποίο παράγονται 101

Δ Η Δομή Δεδομένων του Πίνακα Ιχνοληψίας **103**

Ε *ArrayTracer* : Installation and User's Guide **107**
E.1 Installation Guide 107
E.1.1 Unpack *ArrayTracer* 107
E.1.2 Installing *ArrayTracer* 107
E.1.3 Object Files 108
E.2 *Tracing Tool*: User's Guide 109
E.3 *Analyzing Tool*: User's Guide 115
E.4 *WWW User Interface*: User's Guide 117
E.4.1 *ArrayTracer*'s Main WWW Page 117
E.4.2 Transfer Files WWW Page 117
E.4.3 *Tracing Tool*'s WWW Page 118
E.4.4 *Analyzing Tool*'s WWW Page 119
E.4.5 *Visualization Tool*'s WWW Page 119

ΣΤ Ευρετήριο Ορων **121**

Κατάλογος Πινάκων

3.1 Πίνακας Περιγραφής των χαρακτηριστικών των βασικότερων συστημάτων οπτικοποίησης. 22

4.1 Πίνακας αναγνώρισης ρουτινών ενδοεπικοινωνίας και συγχρονισμού με χρήση των πεδίων του ίχνους επικοινωνίας. 49
4.2 Αντιστοίχιση Καταστάσεων Ανάλυσης σε *PICL* Γεγονότα. 49

5.1 Πίνακας Αντιστοίχισης Γεγονότων σε Χρωματισμούς 65

Κατάλογος Σχημάτων

1.1 Βασικά δομικά στοιχεία της διαδικασίας καθοδήγησης και ιχνοληψίας της εφαρμογής από το ArrayTracer και γραφική αναπαράσταση της ροής δεδομένων ιχνοληψίας (1ο επίπεδο λειτουργίας). 4
1.2 Βασικά δομικά στοιχεία της διαδικασίας ανάλυσης και οπτικοποίησης της εφαρμογής από το ArrayTracer και γραφική αναπαράσταση της ροής δεδομένων (2ο επίπεδο λειτουργίας). 4

3.1 Σύστημα *BALSA* 23
3.2 Σύστημα *TOPSYS* 24
3.3 Εισαγωγή Κώδικα Καθοδήγησης 25
3.4 Ανάλυση και Αναπαράσταση 26

4.1 Βασικά δομικά στοιχεία της διαδικασίας ανάλυσης ιχνών 28
4.2 Παράδειγμα μείωσης του κώδικα καθοδήγησης με χρήση του *ArrayTracer* 30
4.3 *ArrayTracer* vs. *PICL*: Πρώτο πείραμα 32
4.4 *ArrayTracer* vs. *PICL*: Δεύτερο πείραμα 33
4.5 *ArrayTracer* vs. *PICL*: Τρίτο πείραμα 34
4.6 Σχέση Διάταξης *Lamport* 37
4.7 Μηχανισμός ανίχνευσης της ύπαρξης ελλειπούς πληροφορίας. . . . 40
4.8 Περιγραφή του τμήματος Ανάγνωσης Ιχνών 41
4.9 Διαδικασία ανάλυσης σε επίπεδο αποστολής μηνυμάτων. 43
4.10 Διαδικασία ανάλυσης σε επίπεδο παραλαβής μηνυμάτων. 44
4.11 Διαδικασία ανάλυσης σε επίπεδο ομαδοποίησης διεργασιών. 45
4.12 Διαδικασία ανάλυσης σε επίπεδο πηγαίου κώδικα. 46
4.13 Διαδικασία ανάλυσης σε επίπεδο Μνήμης. 47

5.1 *Utilization Count* Διάγραμμα 53
5.2 *Utilization Gannt* Διάγραμμα 53
5.3 *Utilization Summary* Διάγραμμα 54
5.4 *Utilization Meter* Διάγραμμα 54
5.5 *Utilization Profile* Διάγραμμα 55
5.6 *Utilization Kiviat* Διάγραμμα 55
5.7 *Utilization Streak* Διάγραμμα 56
5.8 *Communication Traffic* Διάγραμμα 56
5.9 *Space Time* Διάγραμμα 57
5.10 *Message Queues* Διάγραμμα 57
5.11 *Communication Matrix* Διάγραμμα 58

5.12 *Hypercube* Διάγραμμα 59
5.13 *Network* Διάγραμμα 59
5.14 Color Code Διάγραμμα 60
5.15 *Variables Streak* Διάγραμμα 61
5.16 *Variables Matrix* Διάγραμμα 62
5.17 *Group* Διάγραμμα 62
5.18 *Raw Time* Διάγραμμα 63
5.19 *Loop Iterations* Διάγραμμα 63
5.20 Generation Map Διάγραμμα 64
5.21 *Event Frequency* Διάγραμμα 64
5.22 *Disk Time* Διάγραμμα 65

6.1 Γενεαλογικός Χάρτης των Διεργασιών: Η διεργασία 0 δημιουργεί τις 1, 2, 3, 4, 5, 6, 7 69
6.2 Επικοινωνία μεταξύ των Επεξεργαστών : Τα 4 βασικά βήματα 74
6.3 Κατανομή Προσπελάσεων των Μεταβλητών της εφαρμογής για ένα Παράλληλο Σύστημα με 2, 4, 8, 16 Επεξεργαστές 75
6.4 Χρόνος Εκτέλεσης Διεργασιών *(Communication, Computation, Synchronization time)* σε ένα Παράλληλο Σύστημα με 2, 4, 8, 16 Επεξεργαστές 76
6.5 *Utilization Summary*: Επιμέρους χρησιμοποίηση των Επεξεργαστών σε ένα Παράλληλο Σύστημα με 2, 4, 8, 16 Επεξεργαστές 77
6.6 *Utilization Count*: Ταυτόχρονη χρησιμοποίηση των Επεξεργαστών σε ένα Παράλληλο Σύστημα με 2, 4, 8, 16 Επεξεργαστές 78
6.7 *Utilization Profile*: Διαγράμματα ποσοστού ταυτόχρονης χρησιμο−ποίησης σε ένα Παράλληλο Σύστημα με 2, 4, 8, 16 Επεξεργαστές . . 79
6.8 Απόδοση του παράλληλου αλγόριθμου λείανσης 80
6.9 Βελτίωση της απόδοσης του αλγόριθμου λείανσης 81
6.10 Χρόνοι εκτέλεσης της αρχικής εφαρμογής και της βελτιωμένης . . . 82

7.1 Βασική Σελίδα Διεπαφής με το Χρήστη 84
7.2 Σελίδα Μεταφοράς Αρχείων του Χρήστη 85
7.3 Σελίδα της Διαδικασίας Ιχνοληψίας 86
7.4 Σελίδα της Διαδικασίας Ανάλυσης 87
7.5 Σελίδα της Διαδικασίας Οπτικοποίησησς 88

Α.1 Παράδειγμα εντολών καθορισμού παραμέτρων εισόδου. 93

Β.1 Ροή δεδομένων που σχετίζονται με τον *ArrayTracer* στο *Paragraph* . . 97

Γ.1 Διαδικασία Μετατροπής των ιχνών $Network \Leftrightarrow Host$ Byte Order . . . 99

Δ.1 Γραφική αναπαράσταση της δομής του πίνακα ιχνοληψίας. 103
Δ.2 Οι δομές δεδομένων που χρησιμοποιήθηκαν για την κατασκευή του πίνακα ιχνοληψίας. 105

E.1 Graphical representation of the *trace table* structure. 112
E.2 Structures of lists storing trace parameters. 113
E.3 Array Table structure. 114
E.4 Equivalence Table structure. 114

E.5 *ArrayTracer*'s Main WWW Page 117
E.6 Transfer User's Files WWW Page 118
E.7 *Tracing Tool*'s WWW Page 119
E.8 *Analyzing Tool*'s WWW Page 120
E.9 *Visualization Tool*'s WWW Page 120

Ευχαριστίες

Θα ήθελα να ευχαριστήσω, τον επόπτη καθηγητή μου, κ. Χρήστο Νικολάου, που μου έδοσε τη δυνατότητα να ασχοληθώ με την παρούσα εργασία καθώς και την κ. Καίτη Χούστη, Ευάγγελο Μαρκάτο και Τίτο Σαρειδάκη που με βοήθησαν κατά τη διάρκειά της.

Ακόμη τον συνάδελφο και φίλο Χρήστο Κλουκίνα και να του ευχηθώ καλή συνέχεια.

Εκτός του Πανεπιστημιακού χώρου για την ηθική τους συμπαράσταση, τον Γιώργο Καλαιτζάκη, τον δασκαλό μου Βαγγέλη Κοντοσάκκο και τον Γιάννη Κορτζή.

Τέλος ευχαριστώ τους γονείς μου Παρασκευή και Βασίλη για την υποστήριξη και την υπομονή που έδειξαν.

Κεφάλαιο 1

Εισαγωγή

Η παρούσα εργασία αποτελεί την ολοκλήρωση του *ArrayTracer*, [**?**, **?**] ενός εργαλείου ανάλυσης της επίδοσης παράλληλων εφαρμογών. Η υλοποίηση του εργαλείου χωρίστηκε σε 4 βασικά τμήματα που περιγράφονται περιληπτικά στις επόμενες παραγράφους. Το τμήμα **Ανάλυσης** και **Οπτικοποίησης** αποτελούν το βασικό κομμάτι της υλοποίησης της εργασίας.

- **Στατική Ανάλυση :** Σε αυτό το τμήμα, σχ. 1.1, γίνεται συντακτική ανάλυση του πηγαίου κώδικα της εφαρμογής και εξάγεται η στατική πληροφορία που περιέχει. Ακόμα, επιλέγονται κατάλληλα σημεία για την εισαγωγή εντολών καθοδήγησης, στα οποία εισάγονται κλήσεις σε ρουτίνες της βιβλιοθήκης ιχνοληψίας του εργαλείου μας. Αυτές οι ρουτίνες έχουν σκοπό την παραγωγή των ιχνών που σχετίζονται με το περιβάλλον εκτέλεσης της εφαρμογής (run−time system) και τα οποία δεν μπορούν να παραχθούν κατά την διάρκεια της στατικής ανάλυσης του πηγαίου κώδικα.

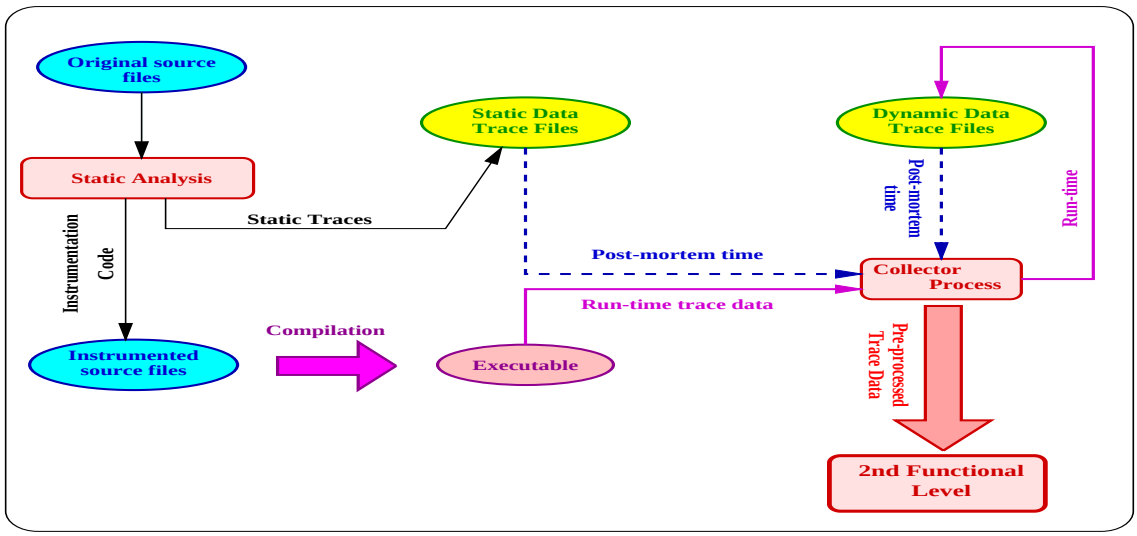
- **Ιχνοληψία :** Σε αυτό το τμήμα, σχ. 1.1, τα ίχνη που παράγονται από την εφαρ−μογή που εκτελείται, συλλέγονται από ειδικές διεργασίες του *ArrayTracer* που ονομάζονται (*συλλέκτες*) (collector processes). Σε αυτήν την φάση εντάσσεται επίσης και η προεπεξεργασία των ιχνών που γίνεται μετά το πέρας της εκτέ−λεσης της εφαρμογής, προκειμένου να επιταχυνθεί η διαδικασία επεξεργασίας τους και η εξαγωγή συμπερασμάτων που αφορούν την απόδοση της εφαρμογής.

- **Ανάλυση :** Κατά την ανάλυση των ιχνών, σχ. 1.2, γίνεται η επεξεργα−σία των ιχνών. Πιο αναλυτικά γίνεται παραγωγή λογικών χρονοσφραγίδων (*timestamps*), δημιουργούνται οι πλήρεις ακολουθίες γεγονότων για κάθε μία από τις διεργασίες, με κριτήριο τη λογική σειρά εμφάνισης των γεγονότων. Τα γεγονότα μεταφράζονται σε καθορισμένης μορφής φόρμες *(PICL)* [1] [**?**] που χρησιμοποιούνται από το τμήμα οπτικοποίησης. Τέλος γίνεται λογικός συγκερασμός όλων των ακολουθιών σε μία, την ακολουθία γεγονότων της παράλληλης εφαρμογής, με κριτήριο την ταξινόμηση κατά αύξουσα σειρά χρονοσφραγίδων.

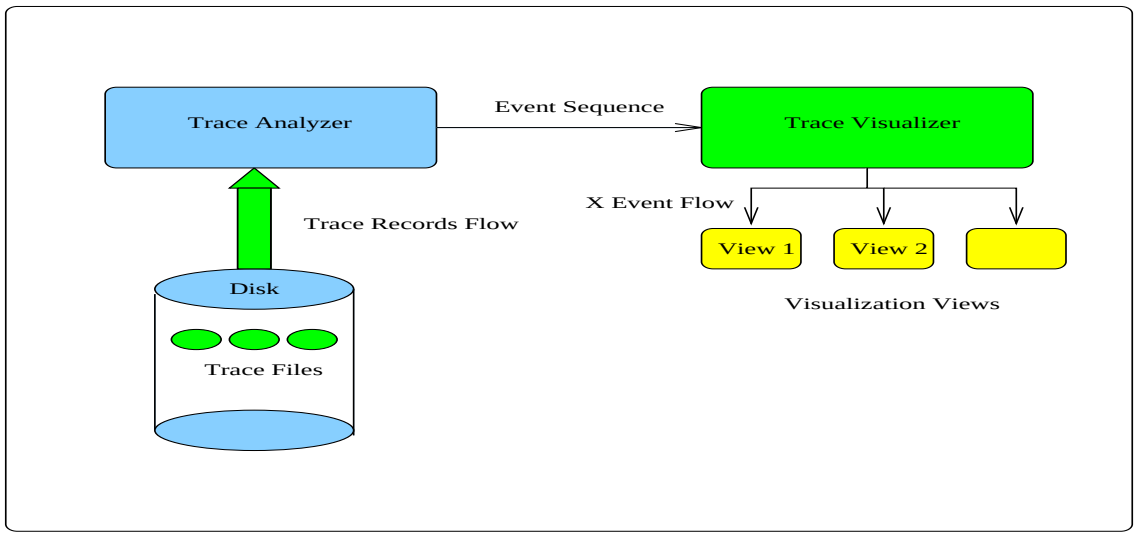
- **Οπτικοποίηση :** Σαν εργαλείο οπτικοποίησης χρησιμοποιήθηκε το *Paragraph*, αφού επεκτάθηκε με γραφικές απεικονίσεις, του τρόπου προσπέλασης της

[1] Το *PICL* είναι μια παράλληλη βιβλιοθήκη επικοινωνίας, με δυνατότητα παραγωγής ιχνών που περιγράφουν τα γεγονότα επικοινωνίας

μνήμης, ομαδοποίησης των διεργασιών, κλήσης συναρτήσεων και υπορουτινών και τέλος απεικονίσεις που σχετίζονται με τις αλλαγές στη ροή εκτέλεσης του προγράμματος. Στο τμήμα αυτό, σχ. 1.2, γίνεται μια προεπεξεργασία της ακολουθίας των *PICL* γεγονότων με σκοπό ορισμένους υπολογισμούς που καθορίζουν τη μορφή των γραφικών απεικονίσεων. Στη συνέχεια τα γεγονότα απεικονίζονται σε οθόνες επιλεγμένες από το χρήστη.



Σχήμα 1.1: Βασικά δομικά στοιχεία της διαδικασίας καθοδήγησης και ιχνοληψίας της εφαρμογής από το ArrayTracer και γραφική αναπαράσταση της ροής δεδομένων ιχνοληψίας (1ο επίπεδο λειτουργίας).



Σχήμα 1.2: Βασικά δομικά στοιχεία της διαδικασίας ανάλυσης και οπτικοποίησης της εφαρμογής από το ArrayTracer και γραφική αναπαράσταση της ροής δεδομένων (2ο επίπεδο λειτουργίας).

Συνοπτικά στα επόμενα κεφάλαια θα ακολουθήσουν τα εξής :

Κεφάλαιο 2 : Θεωρητική Προσέγγιση στη Διαδικασία της Οπτικοποίησης. Μια συνοπτική περιγραφή των ειδών οπτικοποίησης, του σκοπού της οπτικοποίησης και ορισμένων βασικών τεχνικών για τη σωστή σχεδίαση και υλοποίηση ενός εργαλείου οπτικοποίησης.

Κεφάλαιο 3 : Υπάρχον Λογισμικό. Μια συνοπτική περιγραφή διαφόρων ευρέως χρησιμοποιούμενων εργαλείων ανάλυσης και οπτικοποίησης σειριακών και παραλλήλων εφαρμογών.

Κεφάλαιο 4 : Τμήμα Ανάλυσης των Ιχνών. Περιγραφή του τρόπου λειτουργίας του τμήματος ανάλυσης των ιχνών. Περιγραφή της τεχνικής λογικών χρονοσφραγίδων, της τεχνικής ανίχνευσης ελλειπούς πληροφορίας, καθώς και των επιμέρους τμημάτων ανάλυσης.

Κεφάλαιο 5 : Τμήμα Οπτικοποίησης των Ιχνών. Περιγραφή των επεκτάσεων που έγιναν στο *Paragraph*, του τρόπου διασύνδεσης με αυτό.

Παραδείγματα Χρήσης του *ArrayTracer* Μελέτη μιας παράλληλης υλοποίησης του *//ELLPACK*. Μελέτη του αλγορίθμου και χαρακτηρισμός φορτίου κατά την εκτέλεση της εφαρμογής σε ένα παράλληλο σύστημα.

Μελέτη της παραλληλοποιημένης μορφής ένος κλασσικού αλγόριθμου λείανσης.

Κεφάλαιο 7 : Χρήση του *ArrayTracer* **μέσω του Παγκόσμιου Ιστού.** Υλοποίησης ενός συστήματος διεπαφής με το χρήστη για χρήση του εργαλείου μέσω του Παγκόσμιου Ιστού. Μελέτη προβλημάτων ασφάλειας που προκύπτουν.

Κεφάλαιο 2

Θεωρητική Προσέγγιση στη διαδικασία οπτικοποίησης

2.1 Εισαγωγή

Μια εφαρμογή μικρού μεγέθους μπορεί να μελετηθεί με βάση τον πηγαίο κώδικα. Ομως σύγχρονες εφαρμογές αυξημένου μεγέθους και πολυπλοκότητας είναι δύσκολο να μελετηθούν με αυτό τον τρόπο. Επιπλέον η παράλληλη εκτέλεση αυτών των εφαρμογών ενισχύει τη δυσκολία. Η λύση προκύπτει από το γεγονός ότι τα ανθρώπινα όντα χαρακτηρίζονται από την ικανότητα επεξεργασίας μεγάλης ποσότητας πληροφορίας, που περιέχεται σε εικόνες. Παρόλα αυτά όταν η ποσότητα της πληροφορίας αυξάνει σημαντικά, αυξάνει και η δυσκολία του παρατηρητή να επεξεργαστεί την εικόνα. Για αυτό ακριβώς θεωρείται απαραίτητη η ύπαρξη επιπέδων αφαίρεσης στη διαδικασία ανάλυσης που προηγείται λογικά της οπτικοποίησης. Τα επίπεδα αυτά συνιστούν τις πολλαπλές όψεις (*views*) που πρέπει να υπάρχουν σε ένα σύστημα ανάλυσης και οπτικοποίησης εφαρμογών.

2.2 Είδη Οπτικοποίησης

Η διαδικασία της οπτικοποίησης μπορεί να διακριθεί σε τρεις επιμέρους κατηγορίες όσον αφορά το είδος της πληροφορίας που οπτικοποιείται [**?**, **?**].

1. **Οπτικοποίηση Επιστημονικών Δεδομένων** *(Scientific Visualization)* :

   Αυτή η κατηγορία αναφέρεται στη γραφική απεικόνιση δεδομένων που παράγονται από, προσομοιώσεις που εκτελούνται σε υπερυπολογιστές, δορυφόρους, περιφερειακές συσκευές μετρήσεων που χρησιμοποιούνται στην αστρονομία, μετεωρολογία, γεωλογία, ιατρική. Συνεπώς πρόκειται για ένα σημαντικό όγκο πληροφορίας που πρέπει να περιοριστεί, να αναλυθεί και να απεικονιστεί γραφικά [**?**, **?**].

2. **Οπτικοποίηση Προγραμματιστικών Εφαρμογών** *(Program Visualization)*:

   Αυτή η κατηγορία αναφέρεται στην απεικόνιση διαφόρων ενδιαφέρουσων πτυχών της εφαρμογής πχ. το χρόνο εκτέλεσης, την συχνότητα προσπελάσεων στη μνήμη. Περιλαμβάνει τεχνικές που απεικονίζουν την εφαρμογή που αναπαρίσταται με ένα συμβατικό κειμενογραφικό τρόπο *(πηγαίος κώδικας)*, σε εικονογραφημένες παραστάσεις [**?**, **?**, **?**].

Τα συστήματα οπτικοποίησης προγραμματιστικών εφαρμογών αρχικά χρησιμοποιήθηκαν για ανίχνευση λαθών *(debugging)* και για διάφορους εκπαιδευτικούς σκοπούς. Σήμερα χρησιμοποιούνται ευρέως για παρακολούθηση και ρύθμιση της απόδοσης μιας εφαρμογής.

3. **Οπτικοποιημένος Προγραμματισμός** *(Visual Programming)*:

   Οι οπτικοποιημένες γλώσσες προγραμματισμού δίνουν τη δυνατότητα σε μη έμπειρους προγραμματιστές να υλοποιήσουν μια εφαρμογή σε μια γραφική μορφή 2 διαστάσεων. Συνεπώς παρέχεται στους χρήστες η δυνατότητα δημιουργίας πολύπλοκων εφαρμογών με δαπάνη μικρού χρονικού διαστήματος και χωρίς ιδιαίτερη προγραμματιστική εμπειρία. Επίσης οι τεχνικές οπτικοποιημένου προγραμματισμού είναι χρήσιμες για την υλοποίηση φιλικών συστημάτων διεπαφής με το χρήστη *(User Interfaces)*. [**?**, **?**]

2.3 Σκοπός της Οπτικοποίησης

Ο σκοπός της οπτικοποίησης μιας εφαρμογής μπορεί να είναι ένας συνδυασμός των παρακάτω πεδίων μελέτης [**?**, **?**].

- Μελέτη του Αλγορίθμου *(Algorithm Animation)*: Αναπαράσταση των βημάτων του αλγορίθμου, των δεδομένων που επεξεργάζεται, των καταστάσεων από τις οποίες διέρχεται, καθώς και των δομών δεδομένων που χρησιμοποιεί ο αλγόριθμος. Με αυτό τον τρόπο ο χρήστης κατανοεί τι κάνει ο αλγόριθμος, πως δουλεύει και το λόγο για τον οποίο δουλεύει σωστά.

- Μελέτη της συμπεριφοράς της εφαρμογής : Αποκάλυψη διαφόρων ιδιοτήτων της εφαρμογής που δεν είναι προφανής πχ. συχνότητα προσπελάσεων στη μνήμη, διαγράμματα κλήσεων ρουτινών, γράφοι εξάρτησης.

- Μελέτη της απόδοσης της εφαρμογής : Η ανάλυση της απόδοσης μπορεί να ακολουθήσει τις πιο κάτω κατευθύνσεις.

  -- Μετρήσεις : Οι μετρήσεις προυποθέτουν την παραγωγή ιχνών που αντικατοπτρίζουν την απόδοση της εφαρμογής σε σχέση με τη χρησιμοποίηση (*utilization)* του επεξεργαστή, της μνήμης, του δίσκου, του δικτύου.

  -- Μοντελοποίηση: Μελέτη της εκτέλεσης της εφαρμογής με βάση κάποια μοντέλα που προσομοιώνονται από το εργαλείο ανάλυσης και οπτικοποίησης.

  -- Ρύθμιση της απόδοσης: Διάγνωση ύπαρξης προβλημάτων στην απόδοση *(performance bottlenecks)*.

- Ανίχνευση Λαθών : Με την παροχή γραφικών απεικονίσεων των δομών δεδομένων, της ροής των δεδομένων, της ακολουθίας κλήσεων υπορουτινών, της επικοινωνίας μεταξύ διεργασιών και άλλων, επιτυγχάνεται η διάγνωση λογικών σφαλμάτων.

2.4 Χρόνος Οπτικοποίησης

Η διαδικασία ανάλυσης και οπτικοποίησης είναι δυνατόν να πραγματοποιηθεί είτε κατα τη διάρκεια εκτέλεσης της εφαρμογής, είτε μετά το τέλος αυτής. Και οι δύο τεχνικές παρουσιάζουν πλεονεκτήματα και μειονεκτήματα και η επιλογή μιας από αυτές εξαρτάται περισσότερο από το είδος των εφαρμογών στις οποίες απευθύνεται το εργαλείο ανάλυσης και οπτικοποίησης.

2.4.1 Οπτικοποίηση σε πραγματικό χρόνο *(Real Time)*

.

Οπτικοποίηση σε πραγματικό χρόνο σημαίνει ανάλυση και απεικόνιση των χαρακτηριστικών της εφαρμογής, που παρουσιάζουν ιδιαίτερο ενδιαφέρον, κατα τη διάρκεια της εκτέλεσης της εφαρμογής.

Με βάση την τεχνική [**?**, **?**] αυτή προκύπτουν πολύ ακριβής προσομοιώσεις των αλληλεπιδράσεων μεταξύ των διεργασιών [1] της εφαρμογής. Μια προσέγγιση στην οπτικοποίηση σε πραγματικό χρόνο είναι να αποστέλλεται πληροφορία σε ένα εργαλείο οπτικοποίησης σε τακτές χρονικές στιγμές κατά τη διάρκεια εκτέλεσης, μια άλλη είναι η άμεση απεικόνιση της πληροφορίας.

Βασικά μειονεκτήματα στην τεχνική πραγματικού χρόνου είναι η διεύρυνση του χρόνου εκτέλεσης, που είναι αρκετά μεγαλύτερη από αυτή που εισάγει ο κώδικας καθοδήγησης και η έλλειψη επαναληψιμότητας της διαδικασίας οπτικοποίησης, καθώς το περιβάλλον εκτέλεσης συνεχώς μεταβάλλεται πχ. το φορτίο του επε− ξεργαστή. Με Βάση αυτή την παρατήρηση και με δεδομένη την ευαισθησία των παράλληλων εφαρμογών και μη στις συνθήκες του περιβάλλοντος εκτέλεσης.

2.4.2 Οπτικοποίηση μετά το τέλος εκτέλεσης της εφαρμογής *(Post Mortem)*

Η τεχνική αυτή προυποθέτει την ύπαρξη αρχείων αποθήκευσης ιχνών που δημιουρ− γούνται κατα την εκτέλεση της εφαρμογής. Χαρακτηριστικό της μεθόδου είναι ο τρόπος καθορισμού της ακολουθίας των γεγονότων. Αυτός επιτυγχάνεται είτε με παραγωγή πραγματικών χρονοσφραγίδων (τοπικού ή καθολικού χρόνου), είτε με μια διαδικασία παραγωγής λογικών χρονοσφραγίδων. Η δεύτερη μέθοδος στηρίζεται στην αποφυγή παραβίασης λογικών εξαρτήσεων μεταξύ των διεργασιών και παράγει ισοδύναμες οπτικοποιήσεις της εκτέλεσης της εφαρμογής [**?**].

2.5 Στόχοι και τεχνικές σχεδίασης

Οι βασικοί στόχοι που τίθενται κατά τη διαδικασία της σχεδίασης ενός εργαλείου ανάλυσης και οπτικοποίησης της επίδοσης παράλληλων εφαρμογών είναι **ευκολία στην κατανόηση και στη χρήση του εργαλείου και η φορητότητα.** [**?**]. Ενα εργαλείο που πληρεί αυτές τις βασικές προυποθέσεις έχει τη δυνατότητα να είναι ανταγωνιστικό μέσα σε μια πληθώρα άλλων παρόμοιων, καθώς όσο πρωτοποριακό και χρήσιμο να είναι, αν λείπει κάποιο από τα πιο πάνω βασικά στοιχεία, δύσκολα θα υποσκελίσει άλλα, λιγότερο ποιοτικά, εργαλεία.

[1] Ο όρος προσομοίωση χρησιμοποιείται διότι ο επιπλέον χρόνος που απαιτείται για την οπτικοποίηση επηρεάζει την εκτέλεση της εφαρμογής

2.5.1 Ευκολία στην κατανόηση

Βασικός σκοπός της διαδικασίας οπτικοποίησης μιας προγραμματιστικής εφαρμογής είναι η διευκόλυνση της κατανόησης, από την πλευρά του χρήστη, διαφόρων σημαντικών χαρακτηριστικών που την αφορούν, όπως πχ. η επίδοσή της, η συμπεριφορά του αλγορίθμου, η ανίχνευση λογικών σφαλμάτων κ.α.

Συνεπώς είναι απαραίτητο να ισχύουν τα παρακάτω:

- Οι γραφικές απεικονίσεις να είναι συγκεκριμένες και να εξάγονται εύκολα συμπεράσματα από αυτές.
- Τα γραφικά στοιχεία της κάθε απεικόνισης να είναι εκφραστικά και αποτελεσματικά *(effective)*.
- Στην κατασκευή των απεικονίσεων πρέπει να λαμβάνονται υπόψη στοιχεία που προκύπτουν από την καθημερινή ζωή και εμπειρία των χρηστών (πχ. το κόκκινο χρώμα σημαίνει κίνδυνο ή και αναμονή).
- Επιπλέον πρέπει να τηρηθούν περιορισμοί που σχετίζονται με το ανθρώπινο οπτικό και ακουστικό σύστημα εναλλαγών παραστάσεων, όπως το φάσμα των ήχων και των χρωμάτων που μπορεί να διακρίνει ένα ανθρώπινο όν.

2.5.2 Ευκολία στη χρήση

Είναι προφανές ότι ένα καινούργιο εργαλείο, δύσκολο και περίπλοκο στη χρήση του, πολύ δύσκολα θα προσελκύσει τους χρήστες ακόμα και αν διαθέτει πρωτοποριακά στοιχεία που το κάνουν να ξεχωρίζει από άλλα παρεμφερή. Ορισμένες σύγχρονες βασικές προυποθέσεις για ευκολία στη χρήση ενός εργαλείου είναι:

- Πολλαπλά παράθυρα *(Multiwindowing)*.
- Πολλαπλή επεξεργασία *(Multitasking)*.
- Δυνατότητα επέκτασης από το χρήστη.
- Εκσυγχρονισμένο σύστημα διεπαφής με το χρήστη.

2.5.3 Φορητότητα

Απαραίτητη θεωρείται η μη εξάρτηση του όλου συστήματος που σχεδιάζεται από συγκεκριμένο υλικό ή λογισμικό και η ικανότητα επεξεργασίας δεδομένων, που παράχθηκαν από πολλαπλές διαφορετικές αρχιτεκτονικές.

2.6 Βασικές αρχές και προτάσεις για τη σωστή σχεδίαση ενός εργαλείου ανάλυσης και οπτικοποίησης προγραμματιστικών εφαρμογών.

Η σχεδίαση, η υλοποίηση και η χρήση ενός τέτοιου εργαλείου δίνει την εντύπωση δύσκολης και πολύπλοκης διαδικασίας. Παρόλα αυτά υπάρχουν ορισμένες βασικές αρχές που θα πρέπει να τηρηθούν για να υπάρξει ικανοποιητικό αποτέλεσμα [**?**, **?**, **?**].

1. Η ύπαρξη **πολλαπλών επιπέδων αφαίρεσης** στη διαδικασία ανάλυσης είναι απαραίτητη για τη σωστή απεικόνιση και κατανόηση των χαρακτηριστικών της εφαρμογής που μελετάται.

2. Οι ακολουθίες των γεγονότων και οι μεταβολές καταστάσεων από τις οποίες διέρχεται η εφαρμογή πρέπει να αντιστοιχούν σε μεταβολές στις γραφικές απεικονίσεις.

3. Η χρήση **τεχνικών χρωματισμού** είναι απαραίτητη για :

   - Την απεικόνιση της μετάβασης από μια κατάσταση στην επόμενη.
   - Να δοθεί έμφαση σε περιοχές με ιδιαίτερο 1 ενδιαφέρον.
   - Αντιστοίχιση μεγάλων ποσοτήτων σε εναλλαγές χρωμάτων.
   - Συνένωση πολλαπλών οθονών (απεικονίζοντας το αντικείμενο μελέτης με το ίδιο χρώμα σε κάθε οθόνη).

4. Δημιουργία αρχείων στα οποία αποθηκεύεται το ιστορικό των ενεργειών του χρήστη του εργαλείου, με σκοπό την επανάληψη της όλης διαδικασίας.

5. Η ανάλυση της απόδοση της εφαρμογής μπορεί να γίνει τουλάχιστον σε τρία επίπεδα [**?**]:

   - Επίπεδο Εφαρμογής
   - Επίπεδο Αρχιτεκτονικής και Λογικού Δικτύου
   - Επίπεδο Μηχανής και Φυσικού Δικτύου

2.7 Είδη Γραφικών Αναπαραστάσεων

Ακολουθεί μια σύντομη περιγραφή ορισμένων βασικών τρόπων αναπαράστασης δεδομένων.

- Μέτρα *(Meters)* :

  Η απεικόνιση της πληροφορίας γίνεται σε μία διάσταση.

- Γραφικές παραστάσεις σε δύο διαστάσεις *(2D Plots)* :

  Βασικά είδη γραφικών παραστάσεων 2 διαστάσεων είναι:

  -- Γραφικές παραστάσεις γραμμών.
  -- Γραφικές παραστάσεις σημείων.
  -- Γραφικές παραστάσεις επιφανειών.
  -- Ιστογράμματα (απεικόνιση τιμών για ένα συγκεκριμένο αριθμό στιγμών).

- Γραφικές παραστάσεις σε τρείς διαστάσεις *(3D Plots)*

- Chernoff Faces :

  Γίνεται απεικόνιση δεδομένων πολλαπλών διαστάσεων με χρήση εκφράσεων του προσώπου [**?**].

- Γράφοι :

  Τα δομικά τους στοιχεία είναι κόμβοι και ακμές που συνδέουν τους κόμβους μεταξύ τους. Χρησιμοποιούνται για διαγράμματα εξαρτήσεων των δεδομένων της εφαρμογής, διαγράμματα κλήσεων υπορουτινών, για διαγράμματα επικοινωνίας διεργασιών.

- Πίνακες :

  Χρησιμοποιούνται για απεικόνιση των εξαρτήσεων μεταξύ (ίδιων ή διαφορετικών) συνόλων, που αντιστοιχούν στις διαστάσεις του πίνακα, ή απλώς για την απεικόνιση ενός πίνακα δεδομένων.

  Στην πρώτη περίπτωση, κάθε σημαδεμένο στοιχείο (x, y) του πίνακα απεικονίζει τη εξάρτηση μεταξύ των στοιχείων x, y των συνόλων. Στην δεύτερη περίπτωση το χρώμα με το οποίο σημαδεύτηκε το στοιχείο αντιστοιχεί στην τρέχουσα τιμή του αντίστοιχου δεδομένου.

- Διαγράμματα Kiviat : Γεωμετρική απεικόνιση των μεταβλητών ποσοτήτων που μελετούνται. Κάθε μεταβλητή αντιστοιχίζεται σε ένα τόξο κύκλου, σημεία κοντά στο κέντρο του κύκλου αντιστοιχούν σε μεγάλες τιμές της μεταβλητής, ενώ σημεία κοντά στην περιφέρεια αντιστοιχούν σε μικρές τιμές.

Κεφάλαιο 3

Σύνοψη Εργαλείων Ανάλυσης και Οπτικοποίησης

3.1 Εισαγωγή

Με την πάροδο του χρόνου γίνεται όλο και πιο επιτακτική η ανάγκη για ανάπτυξη εργαλείων ανάλυσης και οπτικοποίησης προγραμματιστικών εφαρμογών. Κατα την υλοποίηση τους ακολουθήθηκαν διάφορες κατευθύνσεις που οδήγησαν σε διάφορες κατηγορίες τέτοιων εργαλείων.

Ορισμένα εργαλεία που ακολουθούν τις βασικότερες κατευθύνσεις θα παρουσιαστούν παρακάτω. Τμήματα των εργαλείων αυτών αποτέλεσαν έμπνευση για την υλοποίηση του τμήματος οπτικοποίησης του *ArrayTracer*. Το τμήμα αυτό είναι μια επέκταση του *Paragraph*. Η επιλογή αυτή στηρίχθηκε στο γεγονός ότι είναι προτιμότερη η επέκταση ενός ήδη αναγνωρισμένου εργαλείου το οποίο παρέχει δυνατότητες ανάλυσης στην κατεύθυνση που επιλέξαμε και με την προσθήκη επιπλέον δυνατοτήτων καλύπτει όλο το φάσμα απεικόνισης της πληροφορίας που παρέχει ο *ArrayTracer*. Τα πλεονεκτήματα της επέκτασης ενός υπάρχοντος εργαλείου έναντι στην εξαρχής υλοποίηση του είναι :

- Εξοικονόμηση χρόνου στην υλοποίηση του εργαλείου.
- Νέοι χρήστες προσελκύονται από τις νέες δυνατότητες που παρέχει το εργαλείο μας.
- Οι χρήστες ενός ευρέως γνωστού εργαλείου αποφεύγουν την διαδικασία εκμάθησης και προσαρμογής σε ένα εντελώς νέο εργαλείο.

3.2 Υπάρχον Λογισμικό

3.2.1 *BALSA* I–II

Το σύστημα *BALSA* [**?**], σχ. 3.1, *(Brown Algorithm Simulator and Animator)* σχεδιάστηκε για την οπτικοποίηση σειριακών αλγορίθμων.

Τα βασικά του χαρακτηριστικά είναι :

- Καθορισμός από τον χρήστη των γεγονότων που συγκεντρώνουν ενδιαφέρον.

- Ο πηγαίος κώδικας επαυξάνεται με κλήσεις ρουτινών που επισημαίνουν τα γεγονότα αυτά και προκαλούν μεταβολές σε οθόνες απεικόνισης.
- Ενα σύνολο από αλγορίθμους που αναπαριστούν πληροφορία με διάφορους τρόπους παρέχονται στο χρήστη.
- *Scripts* κατασκευάζονται από το χρήστη που συνδυάζουν αλγόριθμους απεικόνισης.
- Δίνεται η δυνατότητα αποθήκευσης διαφόρων σχηματισμών *(configurations)* με σκοπό την επαναχρησιμοποίηση τους.

Βασική βελτίωση του απόγονου του *BALSA* I, *BALSA* II είναι η πιο εξελιγμένη διαδικασία αποθήκευσης του ιστορικού χρήσης του εργαλείου για τη μελέτη κάποιου αλγορίθμου.

Σύγκριση : Μια από τις βασικότερες διαφορές είναι ότι το σύστημα αυτό σχεδιάστηκε για μελέτη σειριακών εφαρμογών. Αντίθετα ο *ArrayTracer* σχεδιάστηκε με τέτοιο τρόπο ώστε να υπερκαλύπτει την περίπτωση δίνοντας τη δυνατότητα μελέτης παράλληλων εφαρμογών. Επιπλέον το σύστημα αυτό αναλύει την εφαρμογή σε πραγματικό χρόνο. Αντίθετα ο *ArrayTracer* χρησιμοποιεί αρχεία ιχνών. Ειδικότερα ο χρήστης του *BALSA* καθορίζει τα γεγονότα που τον ενδιαφέρουν και για την απεικόνιση πρέπει να γράψει κάποιο επιπλέον κώδικα, ασφαλώς αυτό δεν ισχύει στην περίπτωση του *ArrayTracer*. Η δυνατότητα αποθήκευσης διαφόρων σχηματισμών προσφέρεται και από τα δύο εργαλεία.

Συνολικά το *BALSA* θα μπορούσε να χαρακτηριστεί δευτερεύουσα επιλογή για τη μελέτη παράλληλων εφαρμογών.

3.2.2 *ZEUS*

Το σύστημα *ZEUS* [**?**], χρησιμοποιείται κυρίως για ανίχνευση λογικών λαθών *(debugging)*.

Τα βασικά του χαρακτηριστικά είναι :

- Αυτόματη κατασκευή απεικονίσεων βασισμένων σε ένα σύνολο από γεγονότα που συγκεντρώνουν ενδιαφέρον.
- Ενας προεπεξεργαστής είναι υπεύθυνος για την αποστολή γεγονότων από την εφαρμογή στις οθόνες απεικόνισης. Επιπλέον με τη χρήση του γίνεται και ο καθορισμός των γεγονότων από το χρήστη
- Ενας πίνακας ελέγχου είναι υπεύθυνος για τη τεχνητή δημιουργία γεγονότων.

Σύγκριση : Και στην περίπτωση αυτού του εργαλείου μια βασική διαφορά είναι ότι μελετά τις εφαρμογές σε πραγματικό χρόνο και κυρίως αυτές είναι σειριακές. Επιπλέον παρέχει τη δυνατότητα τεχνητής δημιουργίας γεγονότων που έχουν καθοριστεί από τον χρήστη.

Ο *ArrayTracer* παρέχει κυρίως δυνατότητες μελέτης της απόδοσης μιας παράλληλης εφαρμογής. Παρόλα αυτά όμως υπάρχει δυνατότητα μελέτης του τρόπου εκτέλεσης της εφαρμογής (κλήσεις ρουτινών, προσπέλαση μεταβλητών) για τον έλεγχο

της ορθότητας του προγράμματος. Συνεπώς βασικό πλεονέκτημα του *ArrayTracer* είναι ότι παρέχει πληροφορία για ανίχνευση λαθών, ίσως κάπως περιορισμένη, αλλά υπερτερεί στις πολλές άλλες δυνατότητες που δεν υποστηρίζονται από το *ZEUS*. Βασικό μειονέκτημα του *ZEUS* είναι η σχεδίαση του για σειριακό κώδικα.

3.2.3 *PASTIS*

Το σύστημα [**?**] αυτό χρησιμοποιείται για την οπτικοποίηση της δυναμικής συμπεριφοράς των αλγορίθμων και των δομών δεδομένων της εφαρμογής.

Τα βασικά του χαρακτηριστικά είναι :

- Δεν γίνονται μετατροπές στον πηγαίο κώδικα της υπο μελέτη εφαρμογής. Ανταυτού χρησιμοποιείται ένας ανιχνευτής λογικών σφαλμάτων *(GNU gdb)*.
- *Scripts* χρησιμοποιούνται για την περιγραφή του τρόπου απεικόνισης των δεδομένων που προκύπτουν της εφαρμογής. Τα *Scripts* έχουν παραμέτρους που σχετίζονται τα δεδομένα της εφαρμογής. Επιπλέον ο τρόπος απεικόνισης καθορίζεται από ένα σχεσιακό μοντέλο. Δηλαδή είναι δυνατόν να απεικονιστούν :
  1. Πλειάδες τιμών δεδομένων του ιδίου τύπου[1] *(tuples)*.
  2. Σύνολο από πλειάδες της ίδιας μορφής *(relations)*.
  3. Συσχετίσεις μεταξύ των στοιχείων μιας ή περισσότερων διαφορετικών σχέσεων *(networks)*.

Σύγκριση : Η φιλοσοφία του εργαλείου αυτού δεν ταιριάζει με αυτή του *ArrayTracer* μια που ο βασικός του σκοπός είναι η μελέτη της συμπεριφοράς αλγορίθμων και των δομών δεδομένων τους. Ο *ArrayTracer* παρέχει αυτή τη δυνατότητα σε περιορισμένο βαθμό μια που η παρούσα υλοποίηση χρησιμοποιείται για τη μελέτη εφαρμογών υλοποιημένων σε *Fortran*. Αρα προς το παρόν η δομές δεδομένων που μελετάμε είναι στην πολυπλοκότερη περίπτωση, πολυδιάστατοι πίνακες. Βέβαια μελέτη της απόδοσης της εφαρμογής με βάση το *PASTIS* είναι δύσκολο να γίνει. Παρόλα αυτά η ιδέα της χρήση ενός ανιχνευτή λογικών σφαλμάτων για την παροχή της πληροφορίας είναι πρωτοποριακή.

3.2.4 *Paragraph*

Το σύστημα [**?**] αυτό χρησιμοποιείται για την οπτικοποίηση της επίδοσης παράλληλων εφαρμογών.

Τα βασικά του χαρακτηριστικά είναι:

- Η διαδικασία ανάλυσης γίνεται μετά το τέλος της εκτέλεσης της εφαρμογής *(Post Mortem)*, συνεπώς τα δεδομένα που παράγονται κατά την εκτέλεση αποθηκεύονται σε ένα αρχείο ιχνών ταξινομημένο κατά αύξουσα σειρά χρονοσφραγίδας.
- Αρχικά γίνεται μια προεπεξεργασία για τον προσδιορισμό ορισμένων παραμέτρων κατασκευής των διαγραμμάτων πχ. αριθμός διεργασιών.

[1] Τύποι δεδομένων όπως integer, real, string, boolean

- Στη συνέχεια ακολουθεί η ανάλυση σε επίπεδο :
  - Χρησιμοποίησης Επεξεργαστή.
  - Λογικού Δικτύου.
  - Φυσικού Δικτύου.
  - Αλγορίθμου.
- Γίνεται χρήση τεχνικών χρωματισμού, ενώ υπάρχει και υποστήριξη για μονόχρωμες οθόνες.
- Δυνατότητα επέκτασης από τον χρήστη.

Το *Paragraph σε συνδυασμό με τη παράλληλη βιβλιοθήκη επικοινωνίας* PICL, *[?] είναι ένα εργαλείο της κατηγορίας του* ArrayTracer. *Παρόλα αυτά ο τρόπος που γίνεται η συλλογή των ιχνών με τη χρήση της βιβλιοθήκης* PICL *είναι δαπανηρός τόσο σε χώρο, αλλά και σε χρόνο μια που γίνονται περισσότερες προσπελάσεις στο δίσκο σε σχέση με τον* ArrayTracer.

Το Paragraph είναι το εργαλείο που θα επιλέξουμε για την οπτικοποίηση των ιχνών που παράγει ο *ArrayTracer* διότι είναι πλήρες από πλευράς απεικονίσεων που σχετίζονται με την επικοινωνία των διεργασιών και την χρησιμοποίηση των επεξεργαστών. Επιπλέον η οπτικοποίηση γίνεται μετά το τέλος της εκτέλεσης της εφαρμογής. Τέλος παρέχει τη δυνατότητα επέκτασης, γεγονός που δίνει τη δυνατότητα προσθήκης απεικονίσεων που σχετίζονται με τον τρόπο προσπέλασης της μνήμης και την κλήση υπορουτινών/συναρτήσεων.

3.2.5 *TRACEVIEW*

Το *TRACEVIEW* [?] σχεδιάστηκε σαν ένα γενικής μορφής σύστημα οπτικοποίησης που στηρίζεται στην ανάλυση ιχνών.

Τα βασικά του χαρακτηριστικά είναι:

- Καθορισμός των γεγονότων που ενδιαφέρουν.
- Δυνατότητα επιλογής διάφορων ειδών ανάλυσης και απεικόνισης.
- Δυνατότητα επέκτασης από τον χρήστη.
- Αποτελείται από τρία βασικά τμήματα:
  - Τμήμα διαχείρισης αρχείων ιχνών :
    Το αρχείο αποτελείται από μια *ASCII* επικεφαλίδα, που περιγράφει τη δομή των ιχνών που αποθηκεύονται στο αρχείο, από το σύνολο των ιχνών σε δυαδική μορφή *(binary)*. Το κάθε ίχνος αντιστοιχεί σε ένα γεγονός που αναπαρίσταται γραφικά και η πληροφορία που περιέχει καθορίζεται από το χρήστη.
  - Τμήμα διαχείρισης οθόνης :
    Κάθε οθόνη απεικόνισης αντιστοιχεί σε ξεχωριστό παράθυρο. Επιπλέον κάθε κλάση γεγονότων μπορεί να επιλεγεί και να αναπαρασταθεί σε κάθε μια από τις οθόνες.

-- Τμήμα διαχείρισης αναπαραστάσεων : Η κατασκευή των οθονών απεικόνισης γίνεται από το χρήστη, με βάση ένα δοσμένο σύνολο από γραφικές αναπαραστάσεις. Ετσι οι δυνατές μορφές αναπαράστασης είναι :
  * *Gantt* αναπαραστάσεις.
  * Αναπαραστάσεις συχνοτήτων.
  * Αναπαραστάσεις μέσης τιμής.
  * Αναπαραστάσεις πυκνότητας.

Σύγκριση : Το εργαλείο αυτό παρουσιάζει πολλές ομοιότητες με τον *ArrayTracer*, μελετά παράλληλες εφαρμογές, η μελέτη γίνεται με χρήση αρχείων ιχνών. Ο χρήστης καθορίζει τα γεγονότα που τον ενδιαφέρουν. Το ίδιο γίνεται και από τον *ArrayTracer* με τον καθορισμό των σημείων του κώδικα που θέλει να ιχνοληπτηθούν και με τον καθορισμό του επίπεδου ιχνοληψίας (επικοινωνίας, μεταβλητών, κώδικα). Ενα πλεονέκτημα του *Traceview* είναι η δυνατότητα καθορισμού των περιεχομένων των ιχνών. Αλλά για κάθε γεγονός πρέπει να παράγονται δύο ίχνη, ένα για την είσοδο και ένα για την έξοδο. Αυτό σημαίνει σπατάλη χώρου, πχ. στην περίπτωση γεγονότων προσπέλασης μεταβλητών. Ακόμη οι προσφερόμενες δυνατότητες απεικόνισης κρίνονται περιορισμένες καθώς δεν υπάρχουν διαγράμματα απεικόνισης του τρόπου επικοινωνίας (γράφοι, πίνακες κλπ). Παρόλα αυτά κάθε κλάση γεγονότων μπορεί να απεικονιστεί από κάθε μορφής διάγραμμα.

Γενικά είναι ένα αξιόλογο εργαλείο με αδυναμίες στην απεικόνιση.

3.2.6 *MAP–SHMAP*

Τα συστήματα *MAP* (Memory Access Pattern) και *SHMAP (Shared Memory Access Pattern)* σχεδιάστηκαν για τη μελέτη της απόδοσης της εφαρμογής σε σχέση με τη χρησιμοποίηση της μνήμης του υπολογιστικού συστήματος.

Τα βασικά χαρακτηριστικά τους είναι:

- Καθορισμός ιεραρχίας μνήμης ως εξής :
  -- Κύρια Μνήμη *(Main Memory)*.
  -- Κρυφή Μνήμη *(Cache)*.
  -- Τοπική Μνήμη *(Local Memory)*.
  -- Καταχωρητές *(Vector Registers)*.
- Χρήση αρχείου ιχνών που δημιουργείται κατα την εκτέλεση της εφαρμογής που πρόκειται να μελετηθεί.
- Δυνατότητα κατασκευής διαφόρων σχηματισμών του συστήματος και μελέτη της απόδοσης σε αυτά με βάση τα πραγματικά ίχνη.

Σύγκριση : Το σύστημα αυτό απεικονίζει τον τρόπο προσπέλασης της ιεραρχίας μνήμης των επεξεργαστών ενός παράλληλου συστήματος. Η αδυναμία του δικού μας συστήματος οφείλεται στο γεγονός ότι παράγεται πληροφορία για τον τρόπο προσπέλασης των μεταβλητών και δεν υπάρχει γνώση για το επίπεδο της ιεραρχίας μνήμης στο οποίο γίνονται οι προσπελάσεις (καταχωρητές, κρυφή μνήμη, κύρια μνήμη κλπ). Βέβαια μια επέκταση του *ArrayTracer* θα μπορούσε να περιλαμβάνει

ένα εξομοιωτή προσπέλασης στην ιεραρχία μνήμης. Ακόμη και στην περίπτωση αυτή όμως δεν θα είχαμε την ακριβή συμπεριφορά της εφαρμογής.

Το πλεονέκτημα του *ArrayTracer* είναι ότι παρέχει πληροφορία σχετικά με την επικοινωνία και τη ροή του κώδικα. Αυτά είναι εξίσου σημαντικά για μια παράλληλη εφαρμογή και αγνοούνται από το *MAP*. Συνεπώς το σύστημα υστερεί σε πληρότητα.

3.2.7 *TOPSYS*

Το σύστημα αυτό [**?**], σχ. 3.2, αποτελείται από ένα σύνολο εργαλείων για την ανίχνευση λογικών σφαλμάτων, μελέτη της επίδοσης μιας παράλληλης εφαρμογής, οπτικοποίηση της συμπεριφοράς της και για την προσομοίωση παράλληλων συστημάτων επεξεργασίας.

Τα εργαλεία αυτά είναι :

- *DETOP* (Εργαλείο Ανίχνευσης σφαλμάτων).
- *PATOP* (Εργαλείο Ανάλυσης της Επίδοσης).
- *VISTOP* (Εργαλείο Οπτικοποίησης).
- *SAMTOP* (Προσομοιωτής Παράλληλων Συστημάτων).

Σύγκριση : Το σύστημα αυτό αποτελείται από διαφορετικά εργαλεία και θεωρείται ιδιαίτερα πλήρες. Παρέχει ξεχωριστό εργαλείο ανίχνευσης σφαλμάτων και προσομοίωσης συστημάτων. Επιπλέον παρέχει πληροφορία που σχετίζεται με το υλικό και το λογισμικό. Ο *ArrayTracer* δεν έχει τη δυνατότητα αυτή αφού η εφαρμογή ιχνοληπτείται σε επίπεδο πηγαίου κώδικα. Η αδυναμία του συστήματος αυτού είναι ο φτωχός τρόπος απεικόνισης. Αυτή γίνεται μόνο με ιστογράμματα και διαγράμματα λωρίδων. Αρα πληροφορία που έχει σχέση με τον τρόπο επικοινωνίας δεν είναι δυνατόν να απεικονιστεί. Στο σημείο αυτό υπερτερεί ο *ArrayTracer*. Μια ακόμη βασική διαφορά είναι ότι η εφαρμογή μελετάται σε πραγματικό χρόνο.

3.2.8 *SMILI*

Τα περισσότερα συστήματα οπτικοποίησης περιορίζονται στην απεικόνιση εφαρμογών με μικρό βαθμό παραλληλισμού [2]. Οσο αυξάνει ο βαθμός παραλληλισμού [3] τόσο πιο δύσκολη είναι η κατασκευή των απεικονίσεων και η μελέτη τους από το χρήστη.

Το *SMILI* [**?**] χρησιμοποιεί κινούμενα σχέδια *(cartoon faces)*, γνωστά και ως *Chernoff Faces* [**?**], για την απεικόνιση ενός πεδίου δεδομένων διάστασης p. Συνεπώς το πεδίο των δεδομένων αντιστοιχεί σε ένα σύνολο από *Chernoff Faces* και οι εκφράσεις των προσώπων καθορίζουν την κατάσταση του όλου συστήματος. Τα χαρακτηριστικά του προσώπου αντιστοιχούν σε παραμέτρους μέτρησης πχ. το μήκος της μύτης αντιστοιχεί στο μήκος μιας ουράς εξυπηρέτησης ενός επεξεργαστή.

[2] 8–16 επεξεργαστές

[3] Αριθμός επεξεργαστών πάνω στους οποίους εκτελείται η εφαρμογή

Σύγκριση : Ο τρόπος απεικόνισης παρόλο που είναι ευρηματικός δεν κρίθηκε επαρκής και λεπτομερής καθώς σε εφαρμογές με μικρότερο βαθμό παραλληλισμού απαιτείται μεγαλύτερη λεπτομέρεια στην απεικόνιση.

3.2.9 *Pablo*

Το σύστημα αυτό [**?**, **?**] χρησιμοποιείται για την μελέτη της απόδοσης παράλληλων εφαρμογών και θεωρείται ίσως το πιο πλήρες εργαλείο μελέτης. Το *Pablo* αποτελείται από τα εξής επιμέρους τμήματα.

Λογισμικό εισαγωγής κώδικα καθοδήγησης : Το τμήμα αυτό του εργαλείου, σχ. 3.3 περιλαμβάνει τα εξής τμήματα

- Προεπεξεργαστής C.
- Προεπεξεργαστής FORTRAN.
- Βιβλιοθήκη που παράγει ίχνη.
- Γραφικό σύστημα διεπαφής από το οποίο ο χρήστης καθορίζει τα επιθυμητά σημεία εισαγωγής κώδικα καθοδήγησης.

Ο *ArrayTracer* δεν παρέχει ακόμη τη δυνατότητα μελέτης εφαρμογών σε C, λόγω της απουσίας του αντίστοιχου προεπεξεργαστή. Ακόμη μια αδυναμία του συστήματός μας είναι ότι ο τρόπος καθορισμού των σημείων εισαγωγής του κώδικα καθοδήγησης εξαρτάται από τον κώδικα της εφαρμογής. Αρα αν αυτός αλλάξει πρέπει να επαναληφθεί η διαδικασία. Κατι τέτοιο αποφεύγεται στο *Pablo* με τη βοήθεια του συστήματος διεπαφής.

Λογισμικό ανάλυσης και απεικόνισης των δεδομένων : Στο κομμάτι αυτό του εργαλείου, σχ. 3.4, ο χρήστης καθορίζει μια ακολουθία ανάλυσης και απεικόνισης των δεδομένων. Δηλαδή δημιουργεί ένα κατευθυνόμενο γράφο ανάλυσης και απεικόνισης. Οι κόμβοι του γράφου μπορεί να είναι τριών ειδών:

- Κόμβοι εισόδου/εξόδου
- Κόμβοι επεξεργασίας
- Κόμβοι αναπαράστασης

Οι κόμβοι επεξεργασίας, με βάση την πληροφορία που καθορίζει τη μορφή των ιχνών τα επεξεργάζεται και παράγει πληροφορία χρήσιμη για την απεικόνιση από τους κόμβους αναπαράστασης. Στον *ArrayTracer* ανάλυση και απεικόνιση έχει προκαθορισμένη μορφή. Ο χρήστης έχει τη δυνατότητα να επιλέξει τον τρόπο της απεικόνισης μεταξύ ενός προκαθορισμένου συνόλου απεικονίσεων. Παρόλα αυτά η απλότητα της διαδικασίας αυτής είναι ιδιαίτερα ελκυστική και υποστηρίζεται από την μεγάλη ποικιλία των δεδομένων απεικονίσεων.

Γλώσσα Περιγραφής Ιχνών *SDDF*: Με τη χρήση μιας γραμματικής ο χρήστης μπορεί να καθορίσει τη μορφή που θα έχουν τα ίχνη που θα παράγονται. Δηλαδή εν τέλει την πληροφορία που θα περιέχουν. Αυτή η δυνατότητα δεν παρέχεται από τον *ArrayTracer* διότι ένας από τους στόχους του είναι να κρατηθεί η καθυστέρηση της εφαρμογής σε προβλεπόμενα, χαμηλά επίπεδα.

Αναπαράσταση Δεδομένων με Ηχους: Ασφαλώς αποτελεί μια πολύ πρωτότυπη ιδέα που δεν παρέχεται σαν δυνατότητα από τον *ArrayTracer*.

Σε γενικές γραμμές ο *ArrayTracer* υστερεί σε δυνατότητες και ευελιξία αλλά είναι εξαιρετικά απλός στη χρήση και εύκολος στην εκμάθηση.

3.2.10 *Paradyn*

Το *Paradyn* [**?**] είναι και αυτό ένα εργαλείο ανάλυσης της απόδοσης παραλλήλων εφαρμογών. Βασικός τους στόχος είναι με τη χρήση διαφόρων σχημάτων να μετατρέπουν τη ροή των δεδομένων, που παράγονται από την εφαρμογή, σε γραφικά στοιχεία της μορφής *(X, Y, size, color)*. Ο κώδικας καθοδήγησης τοποθετείται και αφαιρείται κατά τη διάρκεια της εκτέλεσης της εφαρμογής. Ενα τμήμα του εργαλείου χρησιμοποιείται για την αναζήτηση πιθανόν προβλημάτων στην απόδοση που καθορίζονται από το χρήστη.

Σύγκριση: Σε γενικές γραμμές το εργαλείο αυτό είναι παρόμοιο με τον *ArrayTracer* με βασική διαφορά την run−time καθοδήγηση της εφαρμογής που μπορεί να είναι πλεονέκτημα ή μειονέκτημα του εργαλείου ανάλογα με την αύξηση που εισάγει στο χρόνο εκτέλεσης. Επιπλέον η μελέτη της εφαρμογής γίνεται σε πραγματικό χρόνο για την αποφυγή της αποθήκευσης μεγάλων ποσοτήτων πληροφορίας. Επιπλέον το τμήμα αυτόματης ανίχνευσης προβλημάτων απόδοσης θεωρείται καινοτομία και αξιόλογη ιδέα. Μειονέκτημα του συστήματος θεωρείται ότι ο χρήστης πρέπει να γράψει κώδικα για την απεικόνιση των δεδομένων που παράγονται και επιπλέον οι τρόποι απεικόνισης είναι περιορισμένοι (ιστογράμματα, πλέγματα ιστογραμμάτων). Επιπλέον κώδικας πρέπει να προστεθεί και στην εφαρμογή. Ασφαλώς ο χρόνος εκτέλεσης επιβαρύνεται με την προσθήκη κώδικα ανάλυσης και απεικόνισης δεδομένων.

3.2.11 Σύνοψη των υπάρχοντων εργαλείων ανάλυσης και οπτικοποίησης

Στον πίνακα 3.1 περιγράφονται συνοπτικά τα κύρια χαρακτηριστικά των υπάρχοντων εργαλείων ανάλυσης και οπτικοποίησης. Ο χαρακτηρισμός των εργαλείων γίνεται ως προς :

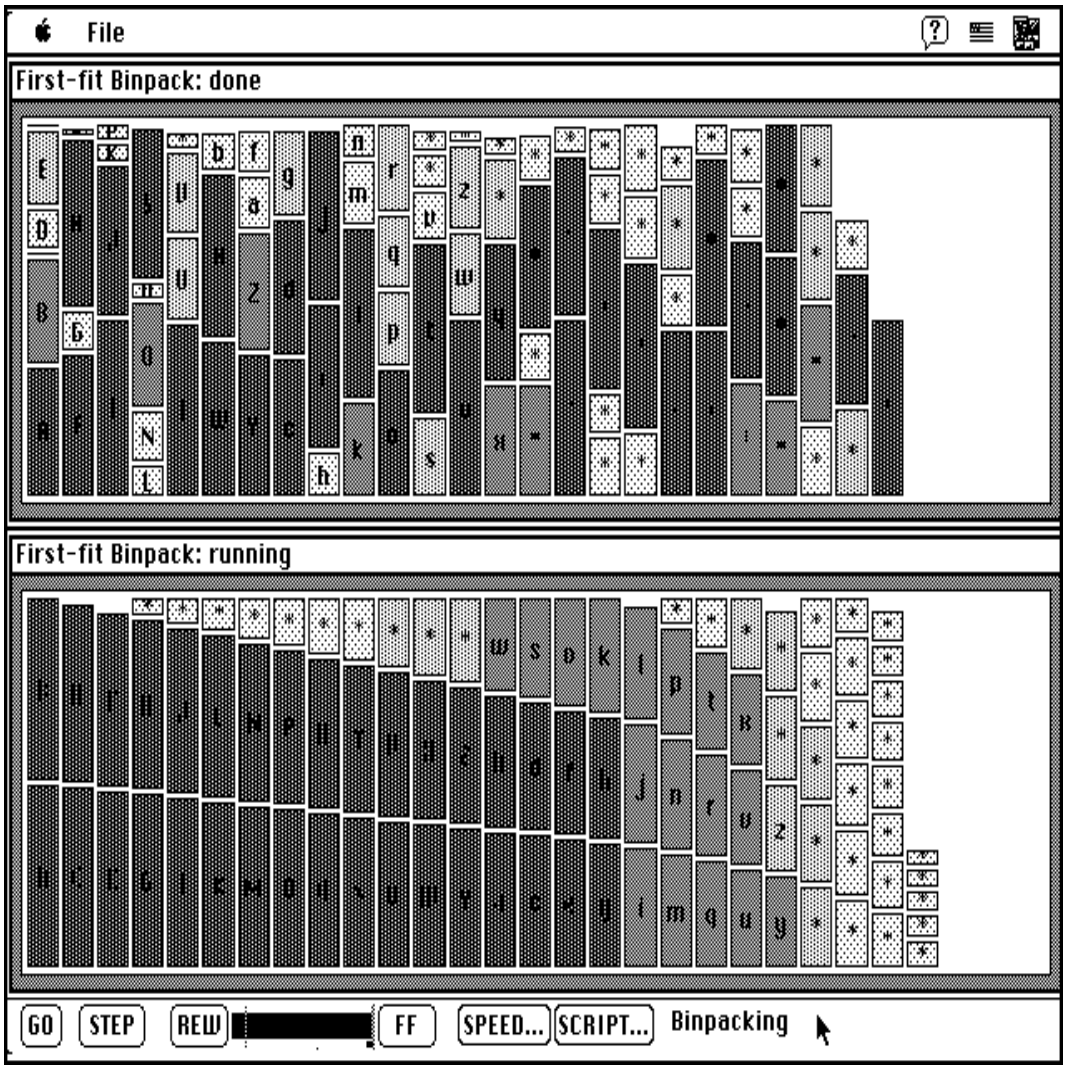
- Το σκοπό της σχεδίασης
  (Purpose : System was designed for **P**arallel Machines).
- Το στόχο του εργαλείου
  (Target : **A**lgorithm Animation,Program **B**ehavior, **P**erformance, **D**ebugging).
- Το χρόνο ανάλυσης
  (Time : **R**eal Time, **P**ost Mortem).

- Το είδος της ανάλυσης

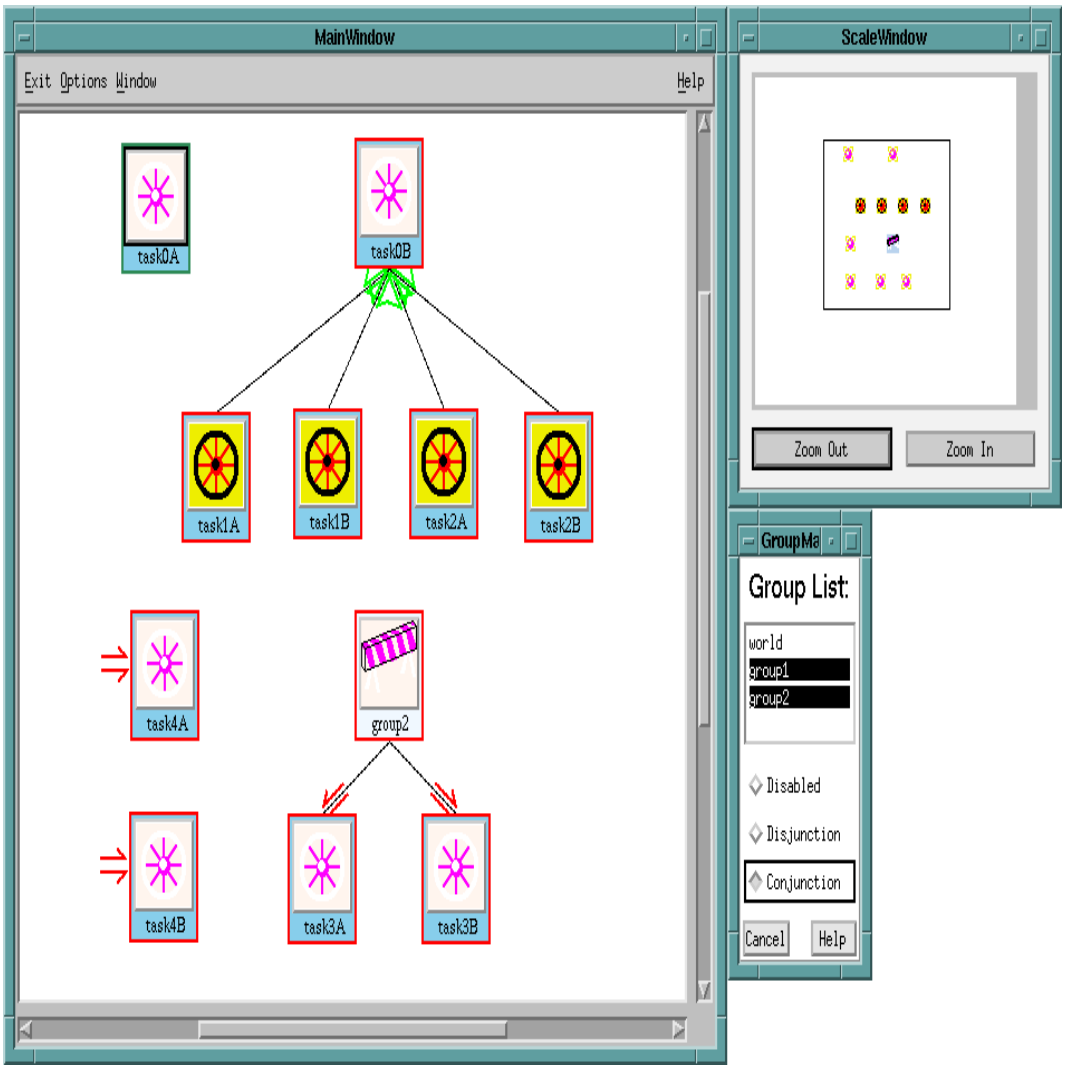
  (Kind : **A**lgorithm, **T**opology, **M**apping, Program **D**ata or **C**ode or **B**ehavior, **P**erformance).

System	Target	Purpose	Time	
BALSA		A	RM	AC
BALSA II		A	R	AC
BUGNET	P	D	R	B
BELVEDER	P	AD	M	B
COMET	P	P	M	P
CLAM		A	M	D
DPM	P	P	RM	P
FAUST		PC	M	CP
IPS−IPS2	P	P	M	MP
IVE	P	B	R	B
IVT		B	R	D
MAKBILAN	P	AD	R	MP
MAP−SHMAP	P	B	M	A
MATRIX	P	P	M	D
PARAGRAPH	P	BP	M	TBM
PARET	P	PC	R	P
PASTIS		A	R	D
PAW−Q+	P	PC	R	B
PAWS	P	PC	R	M
PECAN		AC	M	C
PEGASYS		C	R	C
PIE	P	P	RM	B
POKER	P	CD	R	TD
PPUTTS	P	PD	M	B
PREFACE	P	P	RM	P
PROVIDE		D	R	DB
PV		BC	R	DC
SLALOM	P	P	R	P
SPD−SPM		P	R	P
SMILI		BP	M	P
TANGO	P	AP	M	AP
TASKGRAPHER	P	P	M	TP
TOPSYS	P	PCD	R	DP
TRACEVIEW	P	P	M	P
VIPS		BD	R	D
VISTA	P	D	RM	D
VOYEUR	P	AD	M	AB
ZEUS		A	R	A
PABLO	P	BP	M	TBM
PARADYN	P	BP	R	TBM

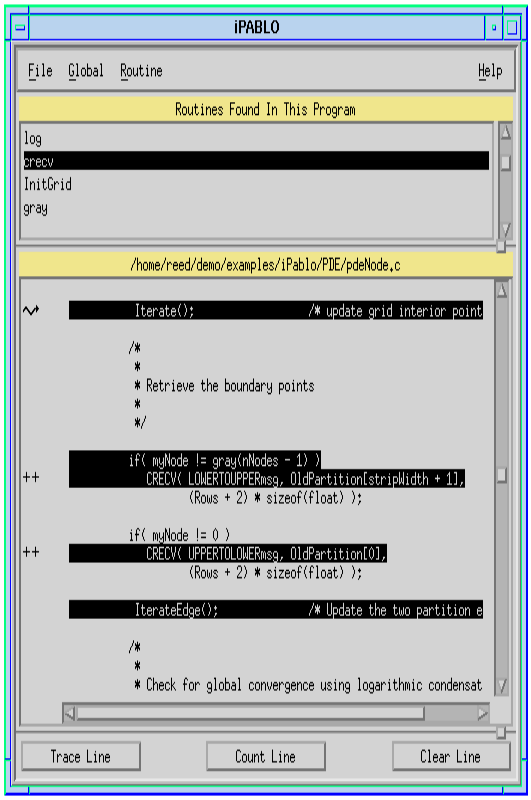
Πίνακας 3.1: Πίνακας Περιγραφής των χαρακτηριστικών των βασικότερων συστημάτων οπτικοποίησης.



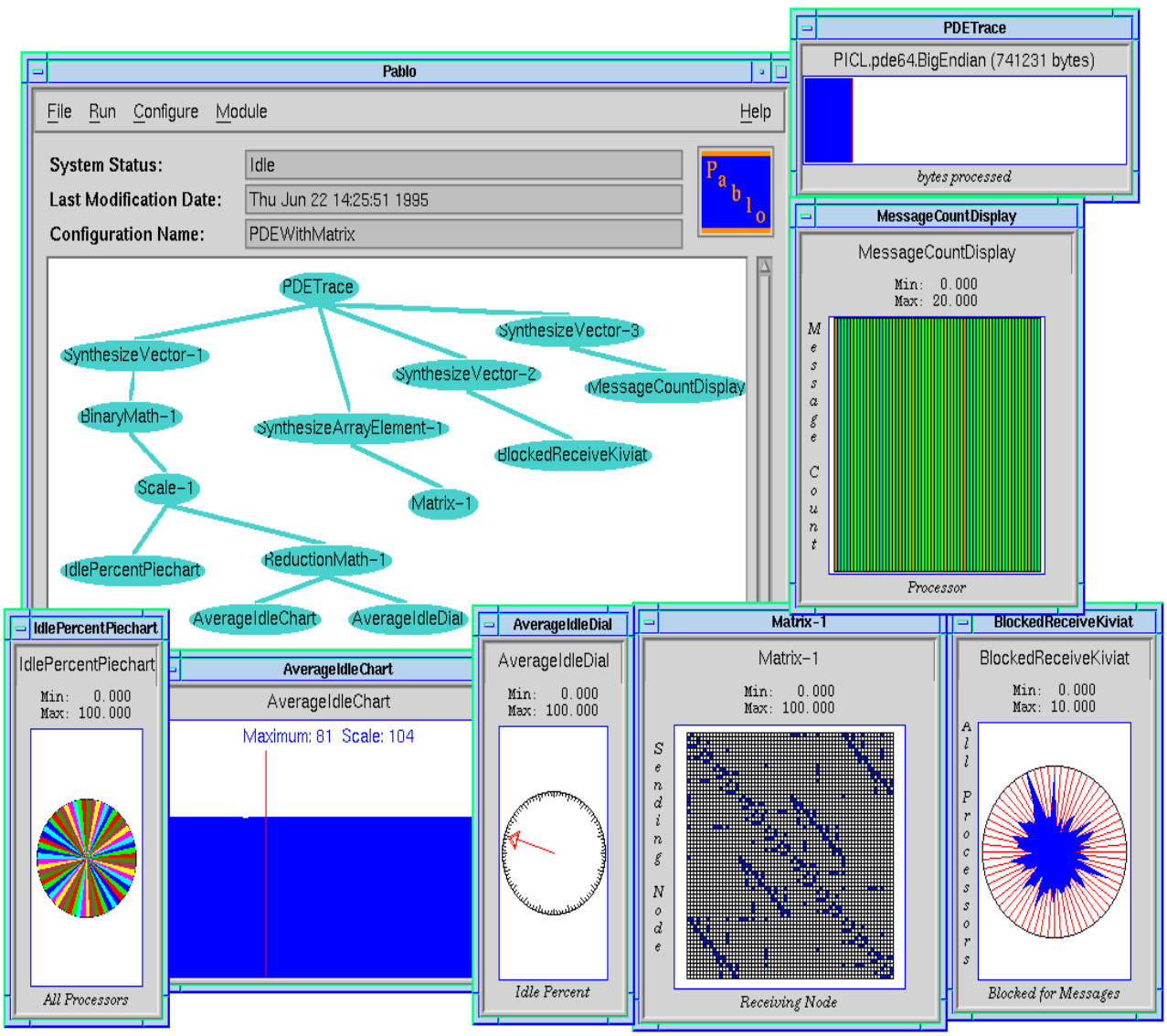
Σχήμα 3.1: Σύστημα *BALSA*



Σχήμα 3.2: Σύστημα *TOPSYS*



Σχήμα 3.3: Εισαγωγή Κώδικα Καθοδήγησης



Σχήμα 3.4: Ανάλυση και Αναπαράσταση

Κεφάλαιο 4

Τμήμα Ανάλυσης των Ιχνών

4.1 Εισαγωγή

Το τμήμα ανάλυσης των ιχνών είναι το τρίτο κατά σειρά δομικό κομμάτι του *ArrayTracer* [**?**]. Αυτό με τη σειρά του αποτελείται από τα επιμέρους τμήματα που απεικονίζονται στο σχήμα 4.1.

- **Τμήμα ανάγνωσης ιχνών** *(Trace Module)* :

  Στο κομμάτι αυτό της διαδικασίας ανάλυσης υλοποιείται η ανάγνωση των ιχνών από τα αρχεία που προέκυψαν κατά την εκτέλεση της εφαρμογής. Τα αρχεία είναι δυνατόν να έχουν παραχθεί από διεργασίες που εκτελέσθηκαν σε διαφορετικές αρχιτεκτονικές [**?**]. Επιπλέον στο τμήμα αυτό γίνονται και ορισμένες αρχικοποιήσεις στη διαδικασία της ανάλυσης.

- **Τμήμα Ανάλυσης Επικοινωνίας** *(Communication Analysis Module)*

- **Τμήμα Ανάλυσης Προσπελάσεων Μεταβλήτών** *(Memory Analysis Module)*

- **Τμήμα ανάλυσης Κώδικα** *(Code Analysis Module)*

- **Γεννήτρια Λογικών Χρονοσφραγίδων** *(Logical Timestamp Generator)*

  Στα πιο πάνω κομμάτια της υλοποίησης πραγματοποιείται η κυρίως διαδικασία ανάλυσης που περιλαμβάνει :

  -- Την παραγωγή μιας ακολουθίας καταστάσεων από τις οποίες διέρχεται μια διεργασία. Με βάση την ακολουθία αυτή θα προκύψει μια ακολουθία γεγονότων *PICL* [**?**, **?**].

  Οι καταστάσεις μπορεί να σχετίζονται, με ίχνη που αφορούν την ενδοδιεργασιακή επικοινωνία, με ίχνη που αφορούν κλήσεις υπορουτινών, συναρτήσεων, εκτέλεση βρόγχων επανάληψης, με ίχνη που αφορούν τις προσπελάσεις που γίνονται στη μνήμη.

  -- Τη διαχείριση ενός συνόλου λογικών ρολογιών, 1 για κάθε διεργασία, με σκοπό την κατασκευή λογικών χρονοσφραγίδων που χρησιμοποιούνται για το συγκερασμό των ακολουθιών γεγονότων, που συνέβησαν κατά την εκτέλεση της κάθε διεργασίας, σε μια καθολική ακολουθία γεγονότων.

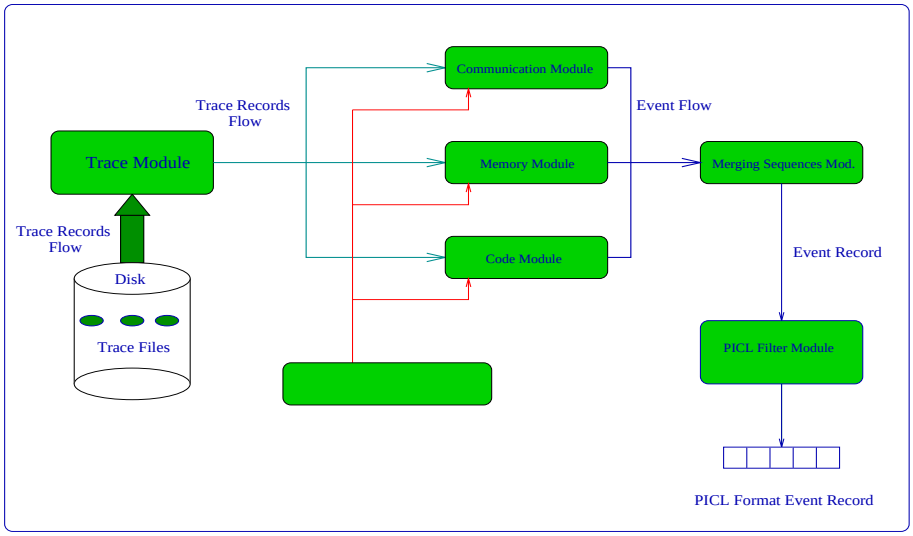
  -- Την εκτέλεση της στατιστικής ανάλυσης των δεδομένων που προκύπτουν από τα ίχνη.

- **Τμήμα Συγκερασμού Ακολουθιών Γεγονότων** *(Merging Sequences Module)*

  Μέχρι αυτό το στάδιο ανάλυσης δημιουργούνται ακολουθίες γεγονότων για κάθε μία από τις διεργασίες που αποτελούν την υπο μελέτη εφαρμογή. Στο κομμάτι αυτό της διαδικασίας ανάλυσης υλοποιείται ο συγκερασμός των επι–μέρους ακολουθιών γεγονότων που προέκυψαν για κάθε μία διεργασία από τα τρία προηγούμενα στάδια ανάλυσης, σε μία ακολουθία γεγονότων. Το κριτή–ριο είναι η ταξινόμηση με αύξουσα σειρά των λογικών χρονοσφραγίδων που δημιουργήθηκαν στα προηγούμενα στάδια ανάλυσης.

- **Φίλτρο παραγωγής γεγονότων σε PICL μορφή** *(Filter Module)*

  Το φίλτρο αυτό παράγει γεγονότα σε συγκεκριμένη μορφή *(PICL Format)* και τα αποστέλει στο τμήμα οπτικοποίησης. Αυτό είναι μια επέκταση του *Paragraph*, ενός ευρέως χρησιμοποιούμενου εργαλείου οπτικοποίησης.



Σχήμα 4.1: Βασικά δομικά στοιχεία της διαδικασίας ανάλυσης ιχνών

4.2 Φόρμες Ιχνοληψίας

Τα ίχνη που παράγονται κατά τη διάρκεια της διαδικασίας ιχνοληψίας είναι δυνατόν να ανήκουν σε μια από της παρακάτω κατηγορίες φορμών ιχνοληψίας.

- Φόρμα ίχνους προσπέλασης μεταβλητής :
  < char **"VAR",** int **TAG,** long **ADDR,** char **OPCODE** >

  – **VAR:** Το πεδίο αυτό καθορίζει ότι το ίχνος αναφέρεται σε μια προσπέλαση μεταβλητής.
  – **TAG:** Το πεδίο αυτό καθορίζει, με βάση τον πίνακα ιχνοληψίας, ποιά μεταβλητή προσπελάστηκε και σε πιο τμήμα κώδικα. Οι πληροφορίες αυτές προέρχονται από τη διαδικασία της στατικής ανάλυσης.

- **ADDR:** Το πεδίο αυτό καθορίζει ποιό στοιχείο πίνακα έχει προσπελαστεί, στην περίπτωση που η μεταβλητή που προσπελάστηκε αναφέρεται σε πίνακα.
- **OPCODE:** Το πεδίο αυτό καθορίζει αν η προσπέλαση στη μνήμη ήταν ανάγνωση ή εγγραφή.

- Φόρμα ίχνους πηγαίου κώδικα :
  < char **"CODE",** int **TAG,** long **TIME** >

  - **CODE:** Το πεδίο αυτό καθορίζει ότι το ίχνος αναφέρεται σε κλήση συνάρτησης, ή υπορουτίνας, ή σε εκτέλεση κάποιου βρόγχου επανάληψης.
  - **TAG:** Το πεδίο αυτό καθορίζει, με βάση τον πίνακα ιχνοληψίας, ποια από τις παραπάνω τρεις περιπτώσεις ισχύει ώστε να ακολουθήσει η αντίστοιχη διαδικασία ανάλυσης.
  - **TIME:**
    * Στην περίπτωση κλήσεις υπορουτίνας ή συνάρτησης το πεδίο αυτό καθορίζει το χρόνο εκτέλεσης της υπορουτίνας ή συνάρτησης.
    * Στην περίπτωση βρόγχου επανάληψης το πεδίο αυτό καθορίζει τον αριθμό των επαναλήψεων.

- Φόρμα ίχνους επικοινωνίας :
  < char **"COMM",** int **TAG,** long **TIME,** int **PID,** char[32] **GRP** >

  - **COMM:** Το πεδίο αυτό καθορίζει ότι το ίχνος αναφέρεται σε κλήση συνάρτησης επικοινωνίας μεταξύ των διεργασιών της υπο μελέτη διεργασίας.
  - **TAG:** Το πεδίο αυτό καθορίζει, με βάση τον πίνακα ιχνοληψίας, αν το ίχνος αναφέρεται σε αποστολή, παραλαβή μηνύματος ή σε κάποιο διαδικασία που αφορά την ομαδοποίηση των διεργασιών.
  - **TIME:** Το πεδίο αυτό καθορίζει το χρόνο εκτέλεσης της διαδικασίας στην οποία αναφέρεται το ίχνος.
  - **PID:** Το πεδίο αυτό καθορίζει την ταυτότητα της διεργασίας που σχετίζεται με το είδος του ίχνους. Σε περιπτώσεις αποστολής/παραλαβής μηνυμάτων αναφέρεται στη διαδικασία που θα παραλάβει/στείλει το μήνυμα.
  - **GRP:** Το πεδίο αυτό περιέχει το όνομα της ομάδας που ανήκει η διεργασία στην οποία αναφέρεται το ίχνος.

4.3 Λογικές Χρονοσφραγίδες

4.3.1 Εισαγωγή: Βασικοί στόχοι του *ArrayTracer*.

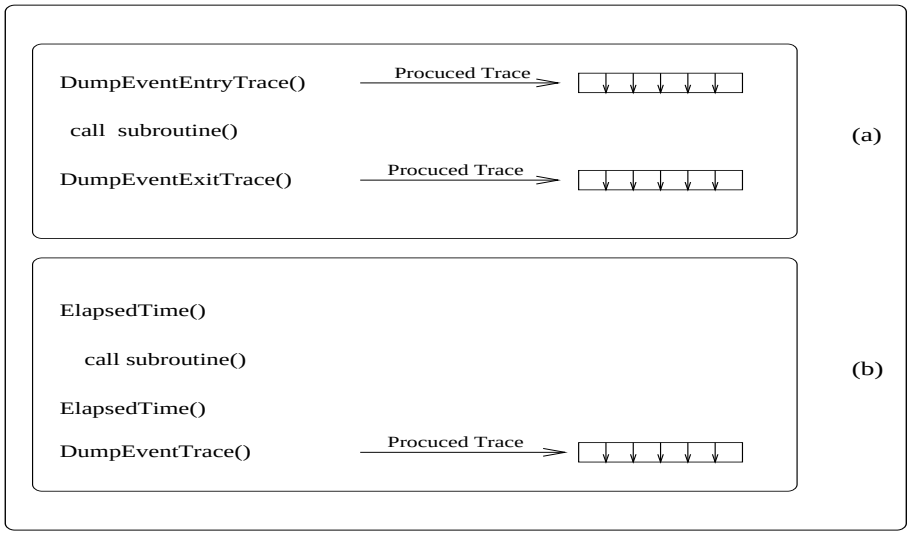
Ενας από τους βασικούς στόχους κατά τη σχεδίαση του *ArrayTracer* ήταν η ελαχιστοποίηση του μεγέθους των αρχείων ιχνών που παράγονται κατά τη διαδικασία ιχνοληψίας και η όσο το δυνατόν μικρότερη διαστολή του χρόνου εκτέλεσης της εφαρμογής μετά την εισαγωγή του κώδικα καθοδήγησης.

Για το λόγο αυτό καθορίστηκε η μορφή φορμών ιχνοληψίας που παρουσιάστηκαν σε προηγούμενη παράγραφο. Τα ίχνη αυτής της μορφής δεν καθορίζουν πλήρως την ακολουθία γεγονότων που λαβαίνουν χώρα κατά την εκτέλεση μιας διεργασίας. Επιπλέον δεν καθορίζεται και η καθολική σειρά των γεγονότων του συνόλου των διεργασιών της υπο μελέτη εφαρμογής.

Ενα παράδειγμα φαίνεται στο σχήμα 4.2.

Στην περίπτωση (a) βλέπουμε τον κώδικα καθοδήγησης που καθορίζει πλήρως ένα γεγονός κλήσης μιας υπορουτίνας. Παρατηρούμε ότι παράγονται δύο ίχνη, το πρώτο καθορίζει την έναρξη του γεγονότος και το δεύτερο το πέρας του.

Στην περίπτωση (b) βλέπουμε τον κώδικα καθοδήγησης που εισάγεται από το δικό μας εργαλείο. Παράγεται ένα μόνο ίχνος που περιγράφει την απόλυτη χρονική διάρκεια του γεγονότος.



Σχήμα 4.2: Παράδειγμα μείωσης του κώδικα καθοδήγησης με χρήση του *ArrayTracer*

Συνεπώς το μέγεθος των ιχνών που παράγονται περιορίζεται στο μισό, ενώ στο μισό περιορίζεται και η διαστολή του πηγαίου κώδικα με συνέπεια την μείωση της διαστολής του χρόνου εκτέλεσης.

Ολα τα παραπάνω προυποθέτουν ότι κατα τη διαδικασία της ανάλυσης θα πρέπει, με χρήση του περιορισμένου αριθμού των ιχνών να καθοριστούν τα εξής.

1. **Η πλήρης ακολουθία γεγονότων** που συμβαίνουν κατά την διάρκεια εκτέλεσης της κάθε διεργασίας της εφαρμογής.

2. **Η καθολική ακολουθία γεγονότων** που προκύπτει με τον συγκερασμό των πιο πάνω ακολουθιών. Η δυσκολία στην περίπτωση αυτή έγκειται στον προσδιορισμό της σχετικής σειράς εκτέλεσης των γεγονότων.

Για την υλοποίηση του πρώτου στόχου, κατά την ανάλυση των ιχνών δημιουργείται μια ακολουθία καταστάσεων, για κάθε διεργασία, που καταλήγει στην παραγωγή γεγονότων μορφής *PICL*.

Για την υλοποίηση του δεύτερου στόχου κατασκευάζονται λογικές χρονοσφραγίδες με χρήση της σχέσης διάταξης *Lamport* [**?**].

Πριν την περιγραφή της τεχνικής αυτής θα μελετήσουμε τον τρόπο που αντιμετωπίζεται το πρόβλημα της σειριοποίησης, καθώς και τη μορφή των ιχνών σε άλλα εργαλεία παρόμοια με τον *ArrayTracer*

ArrayTracer **vs** *PICL*

Καθε γεγονός της παράλληλης βιβλιοθήκης *PICL* [**?**] αντιστοιχεί σε δύο ίχνη που αποθηκεύονται σε κάποιο αρχείο εξόδου. Το πρώτο ίχνος περιγράφει την αρχή του αντίστοιχου γεγονότος και το δεύτερο το πέρας αυτού. Επιπλέον το *PICL* παράγει ίχνη που αντιστοιχούν σε κλήσεις υπολογιστικών συναρτήσεων και δίνει και τη δυνατότητα στο χρήστη να καθορίζει την ιχνοληψία γεγονότων της επιλογής του.

Τέλος τα ίχνη είναι σε *ASCII* μορφή κάτι που επαυξάνει το μέγεθος των αρχείων που παράγονται.

Επιπλέον το *PICL* χρησιμοποιεί το ρολόι συστήματος του κάθε επεξεργαστή που συμμετέχει στο παράλληλο σύστημα, για την παραγωγή της χρονοσφραγίδας του κάθε ίχνους. Τα ρολόγια αυτά περιοδικά συγχρονίζονται με το ρολόι του επεξεργαστή στον οποίο εκτελείται η διεργασία που δημιούργησε όλες τις άλλες. Συνέπεια αυτής της διαδικασίας όμως είναι να επαυξάνεται ο χρόνος εκτέλεσης της εφαρμογής. Ακόμη η σειριοποίηση των ιχνών και η αποθήκευσή τους γίνεται στον επεξεργαστή που ξεκίνησε η εφαρμογή. Για το λόγο αυτό γίνεται ανταλλαγή μηνυμάτων με σκοπό την αποστολή των ιχνών στον επεξεργαστή που ξεκίνησε η εφαρμογή.

Ο *ArrayTracer* εξάγει ένα μόνο ίχνος για κάθε γεγονός, το οποίο περιέχει τη χρονική διάρκεια του γεγονότος, στην περίπτωση που αυτή ενδιαφέρει. Επιπλέον τα ίχνη που παράγονται είναι σε *binary* μορφή. Η σειριοποίηση των ιχνών επιλύνεται όπως αναφέρθηκε χρησιμοποιώντας τις σχέσεις εξάρτησης που ισχύουν μεταξύ των διεργασιών της εφαρμογής. Αναλυτικά η διαδικασία θα περιγραφεί σε επόμενη παράγραφο.

Για να μελετήσουμε τη συμπεριφορά των δύο εργαλείων εκτελέσαμε τα τρία παρακάτω πειράματα :

1ο Πείραμα: Χρησιμοποιήσαμε δύο σειριακές εφαρμογές, ένα αλγόριθμο λείανσης και ένα πυρήνα που εκτελεί πολλαπλασιαμό πινάκων. Για τις δύο αυτές εφαρμογές προγραμματίσαμε το *PICL* να εξάγει ένα ίχνος για κάθε προσπέλαση μεταβλητής. Η μορφή του ίχνους είναι καθορισμένη από το *PICL* για γεγονότα που καθορίζονται από το χρήστη. Το ίχνος περιέχει χρονοσφραγίδα που καθορίζει τη χρονική στιγμή που συνέβει η προσπέλαση. Το αντίστοιχο ίχνος που παράγει ο *ArrayTracer* δεν περιέχει χρονοσφραγίδα. Τα αποτελέσματα της σύγκρισης φαίνονται στο σχ. 4.3.

Συνεπώς παρατηρούμε ότι η καθυστέρηση που εισάγεται από τον *ArrayTracer* είναι πολύ μικρή σε σχέση με αυτή του *PICL*. Η αναλογία του μεγέθους του παραγόμενου αρχείου είναι 1/3.79 για τον *ArrayTracer*. Επιπλέον από τον *ArrayTracer* αποφεύγεται η κλήση του λειτουργικού συστήματος για τον προσδιορισμό της χρονικής στιγμής που συνέβει το κάθε γεγονός.

Βέβαια η δυνατότητα αυτή που προσφέρεται από το *PICL* είναι επιπρόσθετη καθώς ο βασικός του στόχος είναι η ιχνοληψία δεδομένων που σχετίζονται με την επικοινωνία. Για το λόγο αυτό ακολουθούν τα δύο επόμενα πειράματα.