

Distribution Category:
Mathematics and
Computer Science (UC-405)

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, IL 60439

ANL-91/32
Revision 2

Parallel Programming with PCN

by

Ian T. Foster and Steven Tuecke

Mathematics and Computer Science Division

January 1993

This research was supported in part by the National Science Foundation under Contract NSF CCR-8809615, by the Office of Scientific Computing, U.S. Department of Energy under Contract W-31-109-Eng-38, by the Air Force Office for Scientific Research under Contract AFOSR-91-0070, by the Office for Naval Research under Contract ONR-N00014-89-J-3201, and by the Defense Advanced Research Projects Agency under Contract DARPA-N00014-897-K-0745.

Preface

The PCN system is the product of the efforts of many people at Argonne National Laboratory, the California Institute of Technology, and the Aerospace Corporation, including Sharon Brunett, Mani Chandy, Ian Foster, Steve Hammond, Carl Kesselman, Tal Lancaster, Dong Lin, Jan Lindhiem, Robert Olson, Steve Taylor, and Steven Tuecke. The Upshot trace analysis tool was provided by Ewing Lusk. The expanded BNF syntax for PCN was provided by John Thornley. The two-point boundary value application was provided by Steve Wright.

Contents

I	A Tutorial Introduction	1
1	Program Composition	1
1.1	Core Programming Notation	2
1.2	Toolkit	2
1.3	Cross Reference	3
2	Getting Started	4
3	An Example Program	4
3.1	Compiling a Program	5
3.2	Linking a Program	5
3.3	Running a Program	6
3.4	The main() Procedure	7
4	The PCN Language	8
4.1	Concurrent Programming Concepts	8
4.2	PCN Syntax	11
4.3	Sequential Composition and Mutable Variables	13
4.4	Parallel Composition and Definitional Variables	14
4.5	Choice Composition	17
4.6	Definitional Variables as Communication Channels	19
4.7	Specifying Repetitive Actions	20
4.8	Tuples	22
4.9	Stream Communication	26
4.10	Advanced Stream Handling	29
4.11	Interfacing Parallel and Sequential Code	33
4.12	Review	36
5	Programming Examples	36
5.1	List and Tree Manipulation	36
5.2	Quicksort	39
5.3	Two-Point Boundary Value Problem	42
6	Modules	45
7	The C Preprocessor	45
8	Integrating Foreign Code	47
8.1	PCN/Foreign Interface	47
8.2	Compiling with Foreign Code	48
8.3	Linking with Foreign Code	49
8.4	Multilingual Programming	50

8.5	Deficiency of Foreign Interface	50
9	Higher-Order Programs Using Metacalls	50
10	Process Mapping	52
11	Port Arrays	56
12	Reuse of Parallel Code	57
13	Using Multiple Processors	59
14	Debugging PCN Programs	60
14.1	Syntax Errors	60
14.2	Logical Errors	61
14.3	Performance Errors	61
II	Reference Material	63
15	PDB: A Symbolic Debugger for PCN	63
15.1	The PCN to Core PCN Transformation	63
15.2	Obtaining Transformed Code	65
15.3	Naming Processes	66
15.4	Using the Debugger	66
15.5	Examining the State of a Computation	67
15.6	Breakpoints	69
15.7	Debugger Variables	69
15.8	Miscellaneous Commands	71
15.9	Dynamic Loading of .pam Files	72
15.10	Orphan Processes	72
16	The Gauge Execution Profiler	73
16.1	Linking a Program for Profiling	73
16.2	Profile Data Collection	73
16.3	Snapshot Profiles	74
16.4	Data Exploration	74
16.5	The Host Database	75
16.6	X Resources	76
17	The Upshot Trace Analyzer	76
17.1	Instrumenting a Program	77
17.2	Compiling and Linking the Instrumented Program	77
17.3	Collecting a Log	78
17.4	Analyzing a Log	78
18	Standard Libraries	79

18.1 System Utilities	79
18.2 Standard I/O	82
18.2.1 Reference	82
18.2.2 Examples	85
19 Cross-Compiling	88
20 Intel iPSC/860 Specifics	88
21 Intel Touchstone DELTA Specifics	89
22 Sequent Symmetry Specifics	90
23 Network Specifics	90
23.1 Using rsh	90
23.2 Specifying Nodes on the Command Line	91
23.3 Using a PCN Startup File	92
23.4 Starting net-PCN without rsh	93
23.5 Ending a Computation	93
23.6 Limitations of net-PCN	93
24 Further Reading	94
III Advanced Topics	96
25 pcncomp and the PCN linker	96
26 Makefile	96
27 Run-Time System Debugging Options	98
IV Appendices	101
A Obtaining the PCN Software	101
B Supported Machines	102
C Reserved Words	103
D Deprecated and Incompatible Features	104
E Common Questions	105
F PCN Syntax	106

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts



Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research



With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips



Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.