APL 2010 LPA – Berlin
Array **P**rocessing **L**anguages
Learn **P**arallel **A**pplications

Call for Papers

Chairman's Address

Dear friends of APL and array processing languages,

Following the successful APL Berlin 2000 conference we would like to inform you that the global APL2010 conference is taking place again in Berlin from

September 13 to 16, 2010, in Berlin
at the Technical University (TU)

The members of APL Germany and GSE Working Group APL will jointly organise and host this event supported by the GUIDE SHARE Europe (GSE) organisation.

The vendors of APL and Array Processing Systems as well as array-oriented application software and their user groups will be invited to join the conference as sponsors, exhibitors, presenters and participants. Dyalog '10 is already scheduled for taking place integrated into APL2010.

The program committee headed by Dieter Kilsch and Axel Holzmueller is to take care about the APL2010 Conference program and its publications. An APL2010 Conference website is being created.

All APL and array oriented user groups are most heartily invited to join the APL2010 Conference as well as to support it by contributions to the program and to cooperate with the hosting organisations.

We would appreciate very much your taking into account to join APL2010 as well as to provide some contribution. So we would be delighted if you would continue to read this Call for Papers as a very hearty invitation.

We are looking forward to welcoming you in Berlin at APL2010 and honouring you as a speaker.

For the hosting organisations

Gerald Dittrich
Chairman of the APL2010 Organisation Committee

APL2010 Topics

The APL2010 Berlin Conference is seeking contributions which highlight the value and importance of array processing languages in finding solutions to (application) problems and in creating new strategies for programming. How do array processing languages facilitate rapid development, simulations, prototyping, and toolbox creation?

Computers are getting faster and faster, but a real speed-up can only be achieved by exploiting parallel architectures, spreading the workload on several processors in one computer or on several (distributed) computers, maybe worldwide. From their nature array processing languages are supporting the implementation of parallelism. What are their benefits in existing and future parallel HPC environments and what are today's and tomorrow's applications? Where are the limits of parallelism? To which extent (new and existing) array processing applications can automatically be parallelized?

Programming is a constantly evolving discipline. New and challenging applications keep requiring more realistic simulations, stochastic methods, massive data manipulation, and distributed high performance computing. What is the role array processing languages are playing in these areas? How does parallelism help?

Many programming languages and symbolic interpreters, including the different APL implementations, J, K, MatLab, Maple, S, R are becoming stronger array processing languages (in the following referred to as APLs). What features do these languages have in common? How do they differ? Are they converging in terms of primitives, parallelism, functionality, data and control flow, interfaces to hardware and software, and in communication inside one computer and to the outer world? Do these languages miss important data structures or primitives?

The success of a language depends not only on its quality, but also on its development environment - editors, profilers, and debuggers - and on its toolboxes and libraries. These permit people to attack a problem quickly and successfully.

Topics of interest include, but are not restricted to:

- State of the art of array processing languages:
  - Actual situation
  - Forthcoming features
  - Future expectations
- Array processing languages in computer sciences:
  - Parallel Programming
  - (Automatic) parallelization and concurrency
  - Object-oriented programming
  - Meta level programming
  - Interfaces and communication to external environments
  - Artificial intelligence
  - APLs and the World Wide Web

- Challenging (production type) applications solved with array processing languages:
  - Finance and insurance
  - Marketing and ordering
  - R & D HPC calculations
  - Simulation of scientific, technical, economical and social phenomena
  - Control of processes and plants
  - Knowledge acquisition
- Toolboxes for all varieties of applications:
  - Data representation, graphics
  - APLs and (Relational) Data Base Systems
  - Graphical User Interfaces
  - Neural networks, fuzzy logic
  - Automatic parallelization
- Array processing languages in education:
  - Teaching APLs
  - Teaching with APLs
  - APLs in Schools
  - Student Projects
- Theory and practice in array processing language design and implementation:
  - Optimization of APL programs
  - Language design in APLs
  - APLs and irregular data structures
  - Shortcomings of today's APLs

Types of Contributions

Paper:
45 min presentation (+ 15 min discussion)

Short communication:
20 min presentation (+ 10 min discussion)

Tutorial:
90 min interactive session, several sessions may be booked. The main intention should be to get new people interested in APLs.

Workshop:
2 h computer based introduction into an APL language, a tool box or a new technique, several connected or separate sessions may be booked. The main intention should be to get new people interested in APLs.

Poster:
Presentation of achievements, ideas and topics to be discussed.

Birds-of-a-feather:
e.g. national user groups, special purpose groups.

Papers and short communications will be reviewed.

Deadlines

March 15, 2010	Contribution commitment indicating the category of contribution	Author
April 15, 2010	1 page abstract	Author
May 17, 2010	Acceptance notification	Program Committee
June 15, 2010	Preliminary agenda	Program Committee
July 31, 2010	Final version	Author

There are no deadlines for Birds-of-a-feather sessions.

Intent of Submission

We would appreciate very much your contribution to the conference. Your intention to do so should be emailed as soon as you can to contribution@apl2010.de. Technical details for submission layouts will be sent out end of November 2009.

Dieter Kilsch
kilsch@fh-bingen.de

Axel Holzmueller
axel.holzmueller@dpc.de

Program Chair