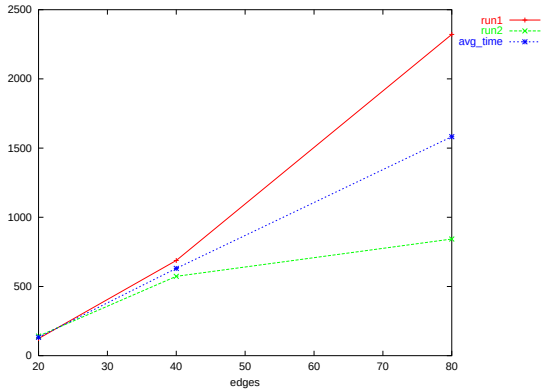
A Tool Set for Computational Experiments

vertices	*edges*	*run1*	*run2*	*avg_time*
10	*20*	*123.6*	*141.3*	*132.4*
20	*80*	*2321.4*	*842.9*	*1582.2*
10	*40*	*432.8*	*832.0*	*632.4*
20	*40*	*943.1*	*314.2*	*628.6*



Susan Hert Lutz Kettner Tobias Polzin Guido Schäfer

MPI für Informatik

The Experimentation Process

do

1. (re)design algorithm
2. (re)implement algorithm
3. gather benchmarking data
4. run benchmark
5. examine results

while (not results_publishable)

The Experimentation Process

do

1. (re)design algorithm
2. (re)implement algorithm $\rightarrow$ use CVS
3. gather benchmarking data
4. run benchmark
5. examine results

while (not results_publishable)

The Experimentation Process

do

1. (re)design algorithm
2. (re)implement algorithm $\rightarrow$ use CVS
3. gather benchmarking data
4. run benchmark $\rightarrow$ record environment
5. examine results

while (not results_publishable)

The Experimentation Process

do

1. (re)design algorithm
2. (re)implement algorithm $\rightarrow$ use CVS
3. gather benchmarking data
4. run benchmark $\rightarrow$ record environment
5. examine results $\rightarrow$ extract data

while (not results_publishable)

The Experimentation Process

do

1. (re)design algorithm
2. (re)implement algorithm $\rightarrow$ use CVS
3. gather benchmarking data
4. run benchmark $\rightarrow$ record environment
5. examine results $\rightarrow$ extract data

while (not results_publishable)

publish

The Experimentation Process

do

1. (re)design algorithm
2. (re)implement algorithm $\rightarrow$ use CVS
3. gather benchmarking data
4. run benchmark $\rightarrow$ record environment
5. examine results $\rightarrow$ extract data

while (not results_publishable)

publish $\rightarrow$ produce graph or table

Unix Tools You Probably (Should) Use

CVS – for source code version control

make – supports experimentation via compilation flags

gnuplot – for producing graphs of data

python, perl, sed, *etc.* – text processing

your favorite editor

- record how, when, and why an experiment was done
- create input files for graphs or tables

Unix Tools You Probably (Should) Use

CVS – for source code version control

make – supports experimentation via compilation flags

gnuplot – for producing graphs of data

python, perl, sed, *etc.* – text processing

your favorite editor

- record how, when, and why an experiment was done
- create input files for graphs or tables

Note: Replacing these tools is not the goal.

Goals

Generally: Provide a comfortable experimentation environment.

Goals

Generally: Provide a comfortable experimentation environment.

Specifically:

- simplify running of computational experiments
- document computational environment automatically
- eliminate some of the tedium involved in processing and analyzing output

Design Principles

- Provide tools that can be used separately and in concert.
- Use file formats easily interpretable by humans.
- Focus on language-independent tools.

Design Principles

- Provide tools that can be used separately and in concert.
- Use file formats easily interpretable by humans.
- Focus on language-independent tools.

Implications

- No restrictions on the kind of experiment
- No restrictions on the output format

Design Principles

- Provide tools that can be used separately and in concert.
- Use file formats easily interpretable by humans.
- Focus on language-independent tools.

Implications

- No restrictions on the kind of experiment
- No restrictions on the output format
- Trying the tools is relatively painless

The Current Tool Set

- **Interface**

Command-line interface with possibility of reading command-line options from a file.

```
labrun --cvs=~/dir --name test -c comment
       --env CXXFLAGS  @optfile
```

- **Configuration Tool**

`labsetup` – creates configuration files for experimental context that remains globally or locally constant

Tools for Running Experiments

`labmex` – makes and executes a program

```
labmex --clean before -m '-DCXXFLAGS=-O3'
       timings 10 20 5
```

`labrun` – runs an experiment and documents its context

```
labrun -n timings -t 10_20_5
       labmex --clean before timings 10 20 5
```

`labrerun` – reruns an experiment given a log file created by a previous run

```
labrerun lab_log/timings-10_20_5.log
```

Output Processing Tools

- Harness the power of Python to be able to manipulate data.
- Use regular expressions and key words to extract data.
- Act as filters by default.
- Use an internal format to achieve many-to-many conversion.

Output Processing Tools

`text2sus` – extracts data from an ASCII file using key words and regular expressions

```
text2sus instance value 'last_val=^final.* (\d+)'
```

`table2sus` – extracts data from an ASCII (`gnuplot`) file

`sus2text` – produces an ASCII (`gnuplot`) file

```
sus2text --sort instance --add same=last_val==value
```

`sus2latex` – produces a simple LaTeX table

`sus2plot` – produces a `gnuplot` graph

`sus2sus` – filter, rearrange, reformat, *etc.*

Features

- Promotes scientific integrity
  - context is documented
  - results are reproducible when possible
- Promotes good programming practice
  - use of source code revision control
  - self-documentation
- No restrictions on the kind of experiment
- No restrictions on the output format

Current Status

- We have tested it and used it.
- Beta release is available from
  `http://www.mpi-sb.mpg.de/~hert/ExpTools`

We want feedback!

- First external release expected beginning of August.

Current Status

- We have tested it and used it.
- Beta release is available from
  `http://www.mpi-sb.mpg.de/~hert/ExpTools`

We want feedback!

- First external release expected beginning of August.

We want feedback!

P.S.

These slides were produced with the document class **prosper**.

`http://prosper.sourceforge.net/`

- written on top of the seminar class
- easily creates slides for either `pdf` or `ps`
- comes with predefined styles
- easy to define your own style

P.S.

These slides were produced with the document class **prosper**.

`http://prosper.sourceforge.net/`

- written on top of the seminar class
- easily creates slides for either `pdf` or `ps`
- comes with predefined styles
- easy to define your own style
- easily supports overlays