How to Add a New Phase in PIPS: the Case of Dead Code Elimination

Ronan KERYELL

Corinne ANCOURT Fabien COELHO Béatrice CREUSILLET

François IRIGOIN Pierre JOUVELOT Julien ZORY

and many other people!

Centre de Recherche en Informatique, École Nationale Supérieure des Mines de Paris

35, rue Saint-Honoré, F-77305 FONTAINEBLEAU CEDEX FRANCE

`http://www.cri.ensmp.fr/pips`

December 11, 1996

CPC'96

Dead code elimination is useful for:

- Specializing codes (e.g. after cloning or inlining)
- Re-engineering old stratified codes
- Cleaning up after source-to-source transformations
- Simplifying before applying some analyses

combined with
partial evaluation, use-def elimination, CFG restructuring

- `OCEAN` example

  25 out of 57 tests removed in the main procedure

  (6 true branches and 19 false branches)

- Dead code elimination & related analyses
- PIPSmake interprocedural consistency manager
- NewGen and PIPSdbm object manager

⇝ PIPS workbench

- 2D BOUSSINESQ fluid layer
- Many options (with or without temperature, etc)
- Initializations in subroutine `START`

$\leadsto$ requires *interprocedural* information to simplify the code

Predicates on integer scalar variables

```
Epips
Buffers Files Tools Edit Search Epips Help
C  P() {NFTVMT==0, NUM1==1, NUSHUF==0, NXACAC==0, NXLOG2==0,
C     NXPRNT==0, NXSCSC==0, NXSHUF==0, NXXCSR==0, NXXOUT==0,
C     NXXRCS==0, NXXXIN==0, TEMPHY:NCALL==0, ZETAPH:NCALL==0}
       CALL START(NX, NY)                                      0006
C  P(KDATA,KPF,KPP,KPRESS,KSM,KTA,KTAPE,KTF,KTP,KUF,KUP,KVF,KVP,KZF,
C     KZP,LMAPP,N1,N2,NOALS,NOTEMP,NUMBER) {KDATA==1, KPF==1,
C     KPP==249, KPRESS==1, KSM==1, KTA==50, KTAPE==0, KTF==1,
C     KTP==50, KUF==1, KUP==249, KVF==1, KVP==249, KZF==1, KZP==249,
C     LMAPP==1, N1==128, N2==128, NFTVMT==0, NOALS==1, NOTEMP==0,
C     NUM1==1, NUMBER==250, NUSHUF==0, NXACAC==0, NXLOG2==0,
C     NXPRNT==0, NXSCSC==0, NXSHUF==0, NXXCSR==0, NXXOUT==0,
C     NXXRCS==0, NXXXIN==0, TEMPHY:NCALL==0, ZETAPH:NCALL==0}
C
C IF  LMAPP = 1 , H  (THE SIZE OF THE BOX IN Y DIRECTION)  SHOULD BE
C SPECIFIED IN THE MAIN PROGRAM.
C
       H = PI                                                  0007
C  P(KDATA,KPF,KPP,KPRESS,KSM,KTA,KTAPE,KTF,KTP,KUF,KUP,KVF,KVP,KZF,
-----Emacs: EPips-0         13:35 Mail    (Fortran EPips)--L229--C3-- 2%----------
```

Use computed preconditions to remove never executed statement

- statements with false preconditions
- true or false branches of tests with constant condition

  compute condition anyway if any side effect*!*
- 0-trip loops
- 1-trip loops

> $\leadsto$ iterative use-def elimination and control flow graph restructuring and partial evaluation

Applying a transformation or asking for an analysis:

¿Which analyses are needed for which module?

¡Just ask for it!

À la `make` demand driven computation

```
suppress_dead_code    > MODULE.code
    < MODULE.code
    < MODULE.proper_effects
    < MODULE.preconditions
```

⇝ extended to cope with interprocedural issues

Interprocedural example

Interprocedurality with CALLERS and CALLEES

```
subroutine B
[...]
C 2 <= M <= 7
L=C(M)
```

```
subroutine A
[...]
C 0 <= I <= 3
K=C(I)
```

```
C 0 <= J <= 7
function C(J)
[...]
```

```
preconditions_inter    > MODULE.preconditions
        < MODULE.code
        < MODULE.transformers
        < CALLERS.preconditions
```

- Preconditions depend on the code
- Dead code elimination depends on the preconditions but modifies the code

$\leadsto$ Possible iterative refinements of the preconditions

Manage the PIPS resources:

- Automaticaly load in memory file resource if necessary
- Architecture and memory location independent
- Read/write NewGen objects
- Non-NewGen objects with user-provided methods
- Basic interface of all PIPS phases
  - Persistence $\leadsto$ multi-run experiences
  - Preemption points at phase boundary
  - Possible external phases

To avoid explicit management of complex data structures:

```
bool suppress_dead_code(string module)
{
  /* get resources from PIPS DBM */
  statement s = db_get_resource(DBR_CODE, module);
  [...]

  gen_recurse(s,                  /* start recursion from */
    statement_domain,             /* domain to visit */
    dead_statement_filter,        /* top-down decision */
    dead_statement_rewrite)       /* bottom-up transformation */

  [...] /* update resources to PIPS DBM */
}
```

NewGen tool

Automatic generation from LaTeX files

Hierarchical internal representation of PIPS:

```
statement = label:entity x number:int x ordering:int
            x comments:string x instruction ;
instruction = sequence + test + loop + goto:statement
            + call + unstructured ;
sequence = statements:statement* ;
```

```
static bool dead_statement_filter(statement s)
{
  instruction i = statement_instruction(s);

  if (!statement_weakly_feasible_p(s))
    /* empty precondition */
    remove_dead_statement(s, i);
  else {
    switch (instruction_tag(i)) {
    case is_instruction_loop:
      loop l = instruction_loop(i);
      ...
    case is_instruction_test:
      test t = instruction_test(i);
      ...
```

- Build complex types from lists, mappings, unions, products...
- Accessors, constructors, destructors, readers, writers...
- Consistency verifyer
- Optimized generic iterators
- Multi-language methods (C & Common-Lisp up to now)
- Support for persistence (PIPSdbm)
- Independent passes can be built (without PIPSmake)

PIPS is based on linear algebra:

– vector, matrix (e.g. Hermite form)

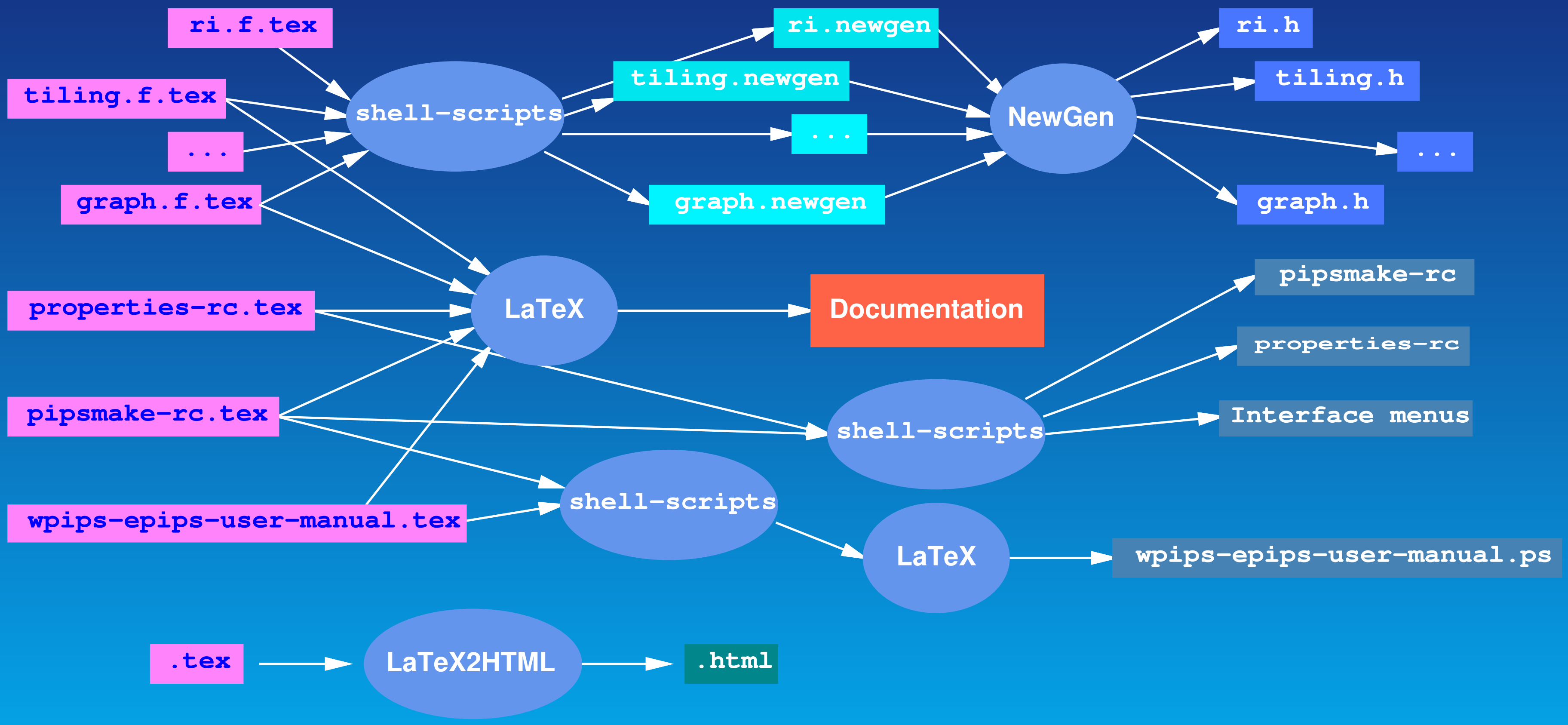
– linear constraints, systems (e.g. Fourier-Motzkin feasibility)

– generating systems, convex hull (Chernikova)

– simplex, PIP

```
bool statement_weakly_feasible_p(statement s)
{
  /* load the precondition system sc associated to s */
  transformer p = load_statement_precondition(s);
  Psysteme sc = predicate_system(transformer_relation(p));
  /* return whether there may be a solution */
  return !sc_empty_p(sc);
}
```

PIPS source consistency is preserved automatically:

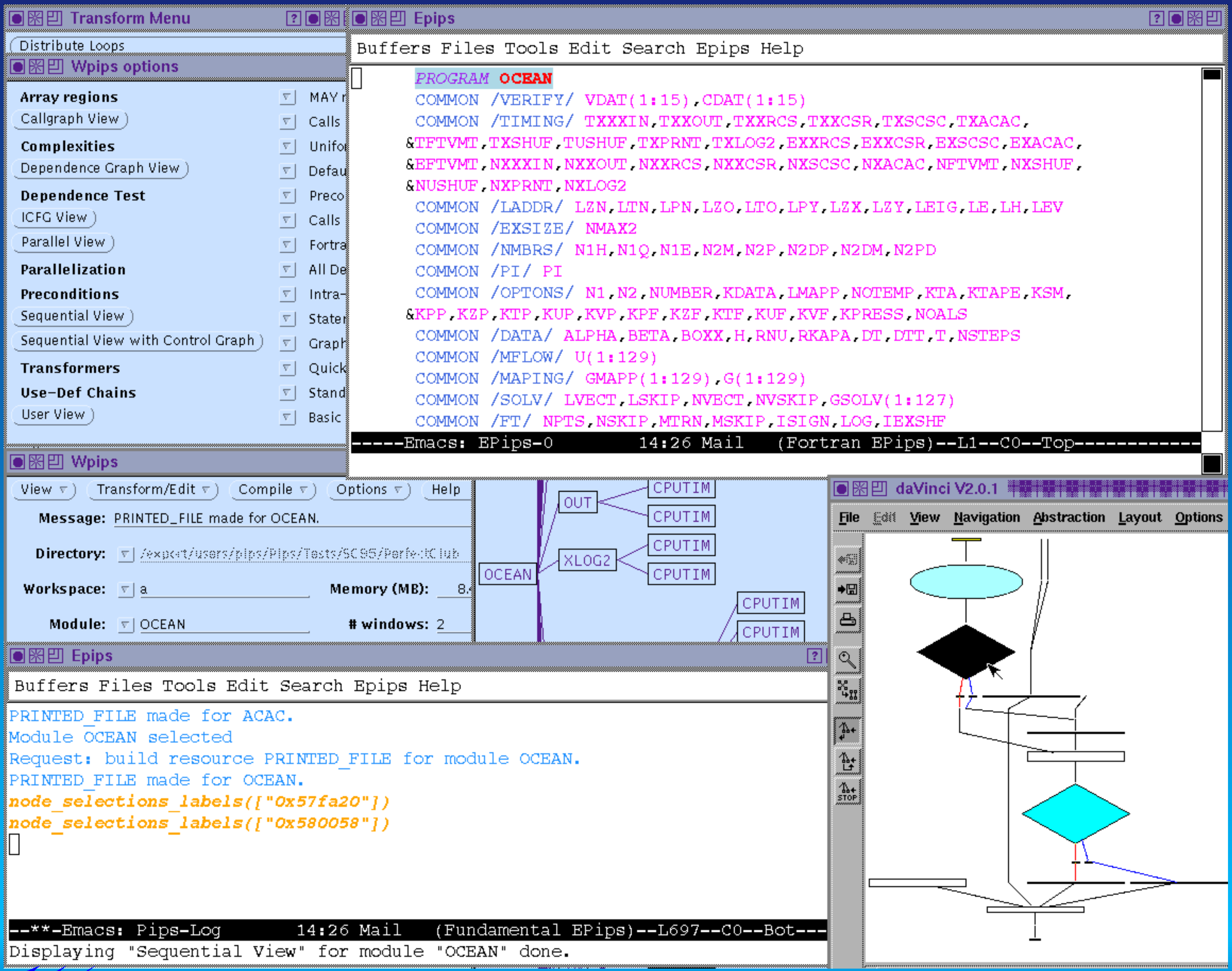


Every file is under SCCS:

development, production and release versions.

```
    \subsubsection{Dead Code Elimination}
    %%@UserManualDocumentation: suppress_dead_code
    Function \verb+suppress_dead_code+ is used to delete non-executed code,
    ...
    %%UserManualDocumentation
\begin{verbatim}
alias suppress_dead_code 'Dead Code Elimination'
suppress_dead_code                  > MODULE.code
        < PROGRAM.entities
        < MODULE.code
        < MODULE.proper_effects
        < MODULE.cumulated_effects
        < MODULE.preconditions
\end{verbatim}
\begin{PipsMenu}{Transformations}
...
  suppress_dead_code
...
\end{PipsMenu}
```

EPips Show



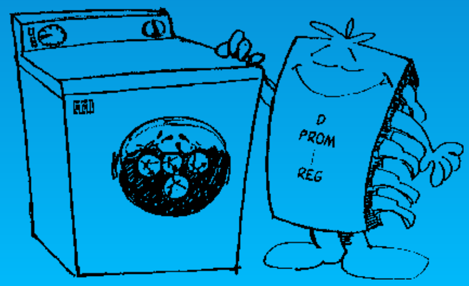
Transform Menu
Distribute Loops
Wpips options
Array regions
Callgraph View
Complexities
Dependence Graph View
Dependence Test
ICFG View
Parallel View
Parallelization
Preconditions
Sequential View
Sequential View with Control Graph
Transformers
Use-Def Chains
User View
Epips
Buffers Files Tools Edit Search Epips Help
PROGRAM OCEAN
COMMON /VERIFY/ VDAT(1:15),CDAT(1:15)
COMMON /TIMING/ TXXXIN,TXXOUT,TXXRCS,TXXCSR,TXSCSC,TXACAC,
&TFTVMT,TXSHUF,TUSHUF,TXPRNT,TXLOG2,EXXRCS,EXXCSR,EXSCSC,EXACAC,
&EFTVMT,NXXXIN,NXXOUT,NXXRCS,NXXCSR,NXSCSC,NXACAC,NFTVMT,NXSHUF,
&NUSHUF,NXPRNT,NXLOG2
COMMON /LADDR/ LZN,LTN,LPN,LZO,LTO,LPY,LZX,LZY,LEIG,LE,LH,LEV
COMMON /EXSIZE/ NMAX2
COMMON /NMBRS/ N1H,N1Q,N1E,N2M,N2P,N2DP,N2DM,N2PD
COMMON /PI/ PI
COMMON /OPTONS/ N1,N2,NUMBER,KDATA,LMAPP,NOTEMP,KTA,KTAPE,KSM,
&KPP,KZP,KTP,KUP,KVP,KPF,KZF,KTF,KUF,KVF,KPRESS,NOALS
COMMON /DATA/ ALPHA,BETA,BOXX,H,RNU,RKAPA,DT,DTT,T,NSTEPS
COMMON /MFLOW/ U(1:129)
COMMON /MAPING/ GMAPP(1:129),G(1:129)
COMMON /SOLV/ LVECT,LSKIP,NVECT,NVSKIP,GSOLV(1:127)
COMMON /FT/ NPTS,NSKIP,MTRN,MSKIP,ISIGN,LOG,IEXSHF
-----Emacs: EPips-0 14:26 Mail (Fortran EPips)--L1--C0--Top------------
Wpips
View Transform/Edit Compile Options Help
Message: PRINTED_FILE made for OCEAN.
Directory:
Workspace: a
Memory (MB):
Module: OCEAN
windows: 2
OUT
XLOG2
OCEAN
CPUTIM
daVinci V2.0.1
File Edit View Navigation Abstraction Layout Options
Epips
Buffers Files Tools Edit Search Epips Help
PRINTED_FILE made for ACAC.
Module OCEAN selected
Request: build resource PRINTED_FILE for module OCEAN.
PRINTED_FILE made for OCEAN.
node_selections_labels(["0x57fa20"])
node_selections_labels(["0x580058"])
--**-Emacs: Pips-Log 14:26 Mail (Fundamental EPips)--L697--C0--Bot---
Displaying "Sequential View" for module "OCEAN" done.

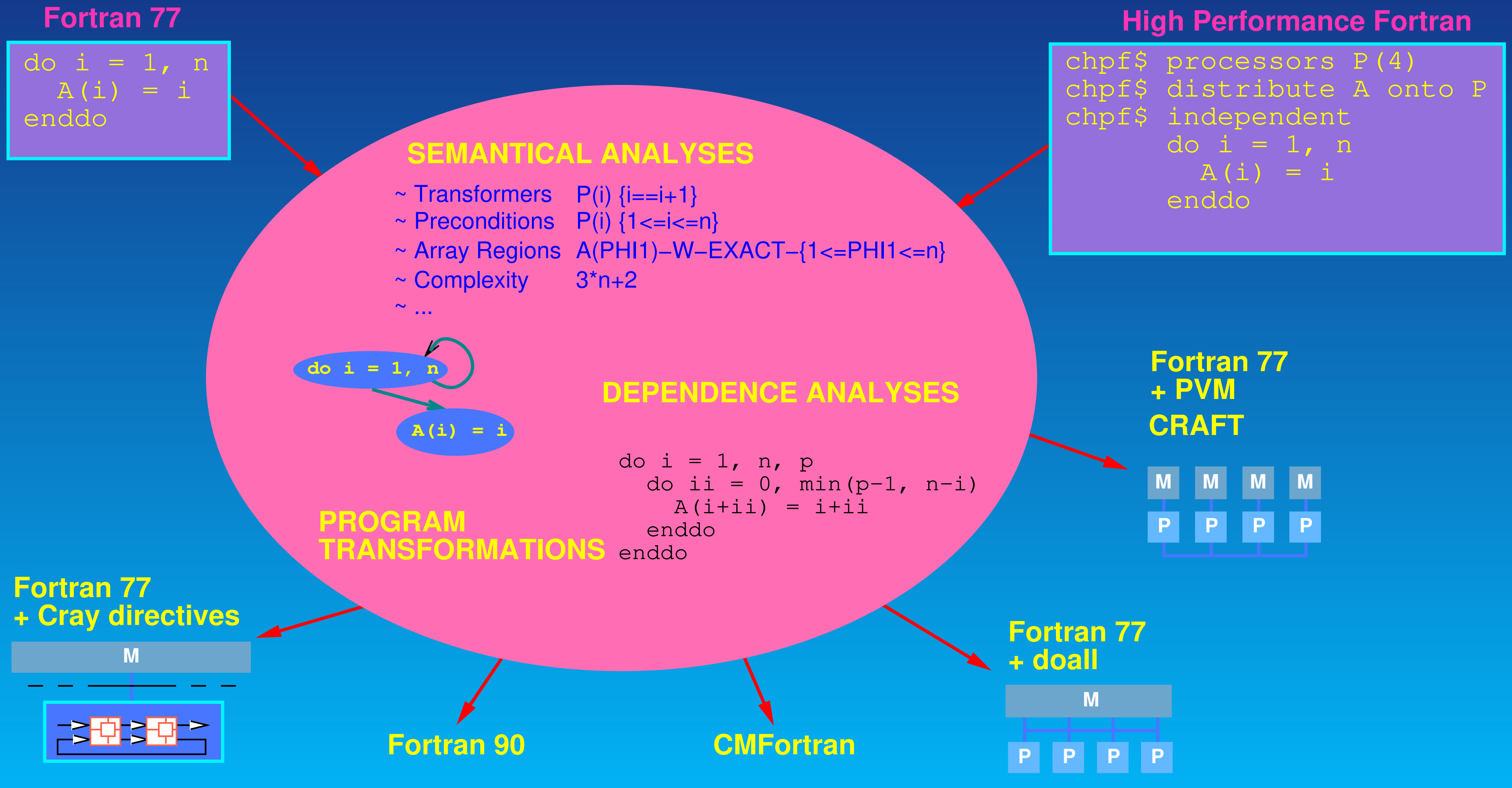
The little *cuisine*:

Cassolette de PIPS aux phases

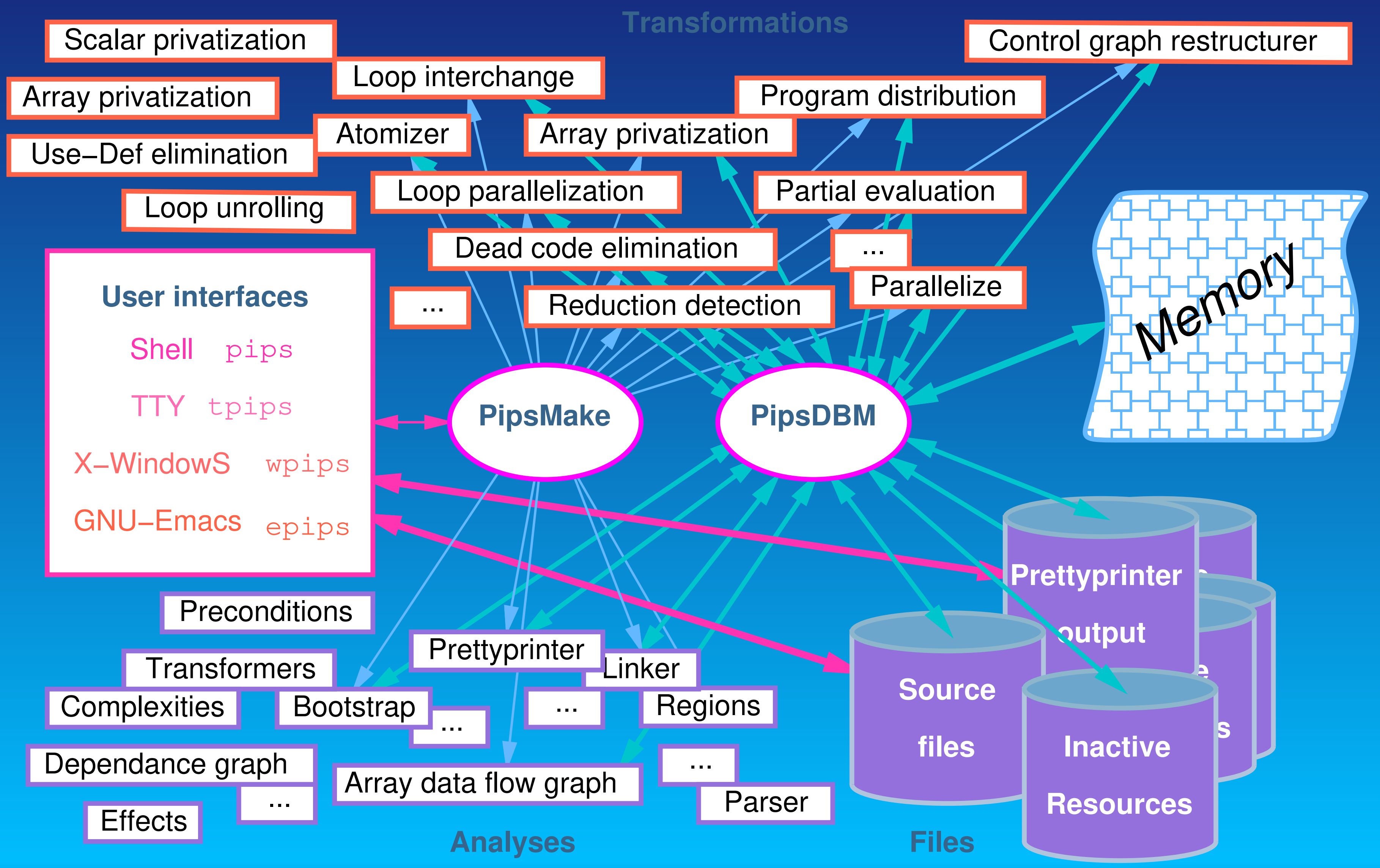
- Create NewGen objects if necessary
- Write your code(!) to deal with one module
- Compile, link and test with `pips-makemake`
- Declare the new phase in `pipsmake-rc.tex`

⇝ New interprocedural analysis/transformation with graphical interface





Fortran 77
do i = 1, n
A(i) = i
enddo
High Performance Fortran
chpf$ processors P(4)
chpf$ distribute A onto P
chpf$ independent
do i = 1, n
A(i) = i
enddo
SEMANTICAL ANALYSES
~ Transformers P(i) {i==i+1}
~ Preconditions P(i) {1<=i<=n}
~ Array Regions A(PHI1)–W–EXACT–{1<=PHI1<=n}
~ Complexity 3*n+2
~ ...
do i = 1, n
A(i) = i
DEPENDENCE ANALYSES
do i = 1, n, p
do ii = 0, min(p-1, n-i)
A(i+ii) = i+ii
enddo
enddo
PROGRAM TRANSFORMATIONS
Fortran 77
+ PVM
CRAFT
M M M M
P P P P
Fortran 77
+ Cray directives
M
Fortran 90
CMFortran
Fortran 77
+ doall
M
P P P P


Transformations
Scalar privatization
Loop interchange
Control graph restructurer
Array privatization
Program distribution
Atomizer
Array privatization
Use−Def elimination
Loop parallelization
Partial evaluation
Loop unrolling
Dead code elimination
...
Parallelize
...
Reduction detection
Memory
User interfaces
Shell pips
TTY tpips
X−WindowS wpips
GNU−Emacs epips
PipsMake
PipsDBM
Prettyprinter output
Source files
Inactive Resources
Preconditions
Prettyprinter
Transformers
Linker
Complexities
Bootstrap
...
...
Regions
Dependance graph
...
Array data flow graph
...
Effects
Parser
Analyses
Files

Other compilation workbenches...

- SUIF (Monica Lam, Stanford)
  - rather C language oriented, + low level code generation
  - no interprocedural features in the released version
  - cannot deal with Fortran `EQUIVALENCE`
  - no PIPSmake equivalent to automatically enforce coherence and deal with interproceduralitу
  - no NewGen like object manager $\rightsquigarrow$ write your own C++ classes

- PARAFRASE 2 (Constantin Polychronopoulos, UIUC)
  - no PIPSMAKE equivalent (- interprocedurality)
  - no NEWGEN like object manager (raw C++)
- POLARIS (David Padua, UIUC)
  - + Fortran semantical consistency checker
  - no PIPSMAKE equivalent (- interprocedurality)
  - no NEWGEN like object manager (raw C++)

- PIPS = interprocedurality + semantics + linear programming
- Advanced analyses and transformations
- Designed as an open workbench:
  - Independent developments at CEA
  - Quick prototyping of new phases (new regions at Southampton U., array privatization in 2 days at CRI)
  - Good platform for studying interprocedurality
  - HPFC for PVM, Fortran for SIMD and T3D at CEA
- 8 year old and still alive... Available for Solaris 1.1.1 and Solaris 2.5 at `http://www.cri.ensmp.fr/pips`