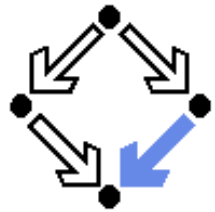
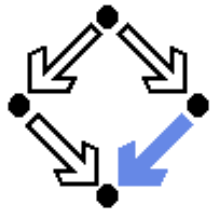
The RISC ProofNavigator
Third Status Report

Wolfgang Schreiner
Wolfgang.Schreiner@risc.uni-linz.ac.at

Research Institute for Symbolic Computation (RISC)
Johannes Kepler University, Linz, Austria
http://www.risc.uni-linz.ac.at

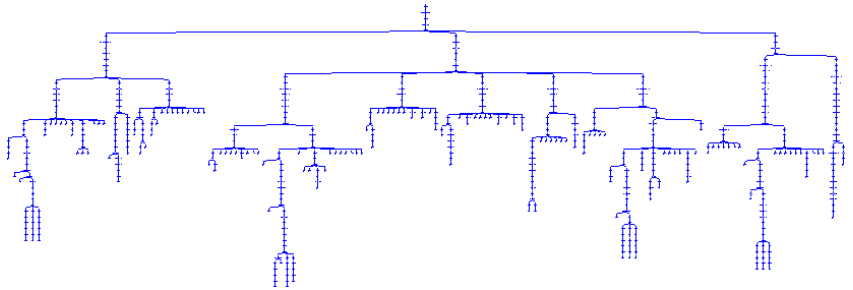


Outlook in March 2006



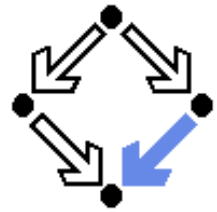
Some more refinements on version 0.52.

- Automatic instantiation before proof state simplification.
  - Instantiate universal assumptions and existential goals.
  - Instantiation values are those ground terms that occur in proof state.
  - Will cut proof trees that only depend on trivial instantiations.
- Larger evaluation example.
  - Client/server system.
- User and reference manual.



Release in the second half of the semester.

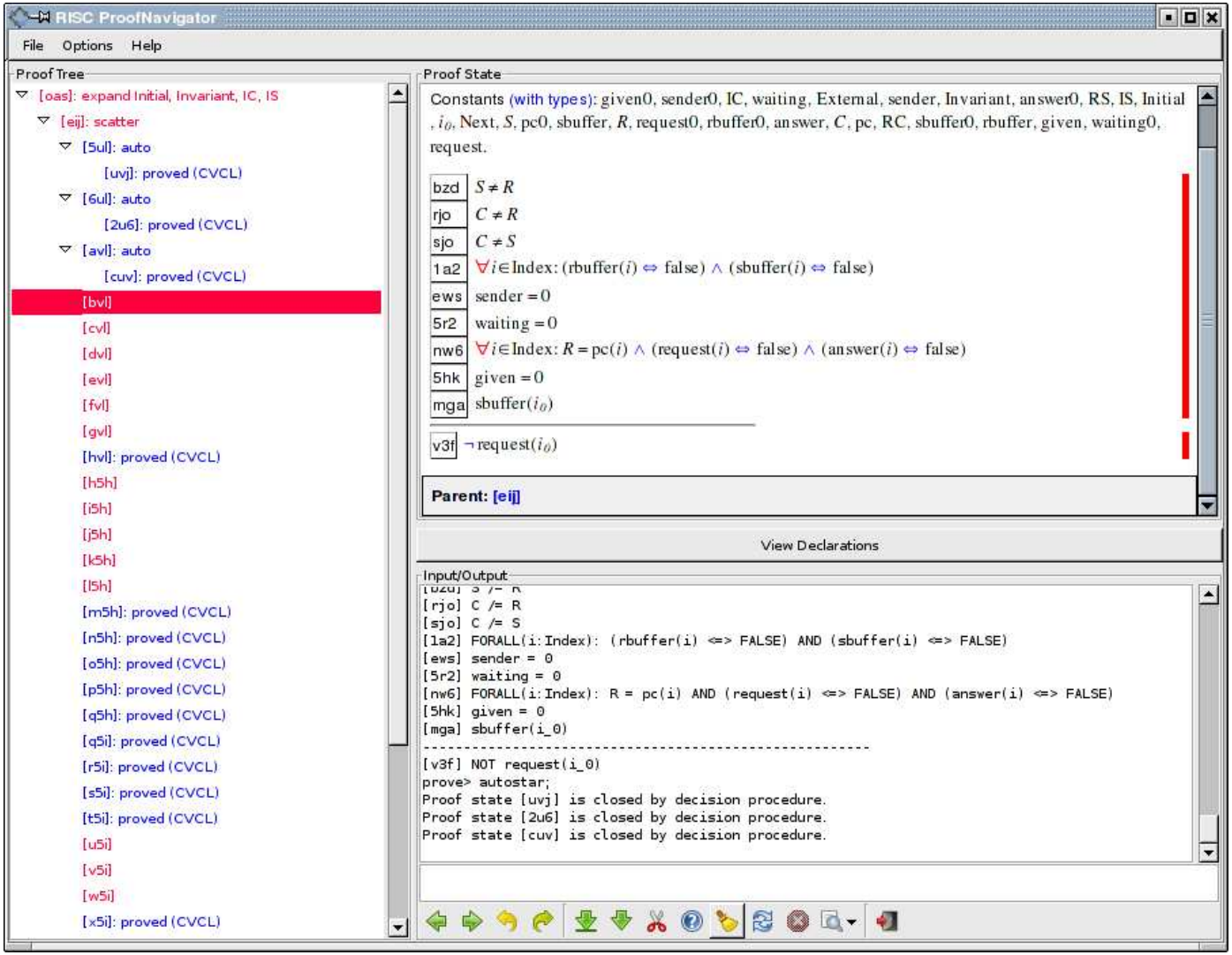
State in June 2006



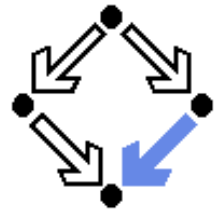
- Automatic instantiation implemented.
  - `auto` command for automatic instantantiation.
    - Replaces `simplify` (automatic CVCL instantiation).
  - `autostar` repeats `auto` also in sibling states.
    - Large degree of automation can be applied to proof.
  - Only at explicit request because potentially costly.
- Client/server example successfully verified.
  - Turned out to be much simpler than original PVS solution.
- Refinements and bug fixes.
  - Now very reliable (not aware of further bugs).

Pre-release version 0.9.

ProofNavigator GUI



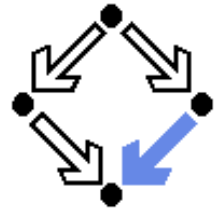
Proof Examples



- Induction proofs: `sum` and `sum0`.
- Quantifier proofs: `quant` and `quant0`.
- Axiomatization of arrays: `arrays1`.
- Constructive definition of arrays: `arrays2`.
- Linear search: `linsearch`.
- Binary search: `binarysearch`.
- Client/Server system: `clientServer`.

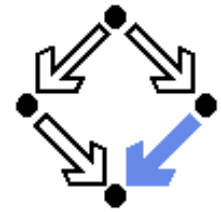
Comparison with previous ProofNavigator version.

Axiomatization of Arrays (Original)

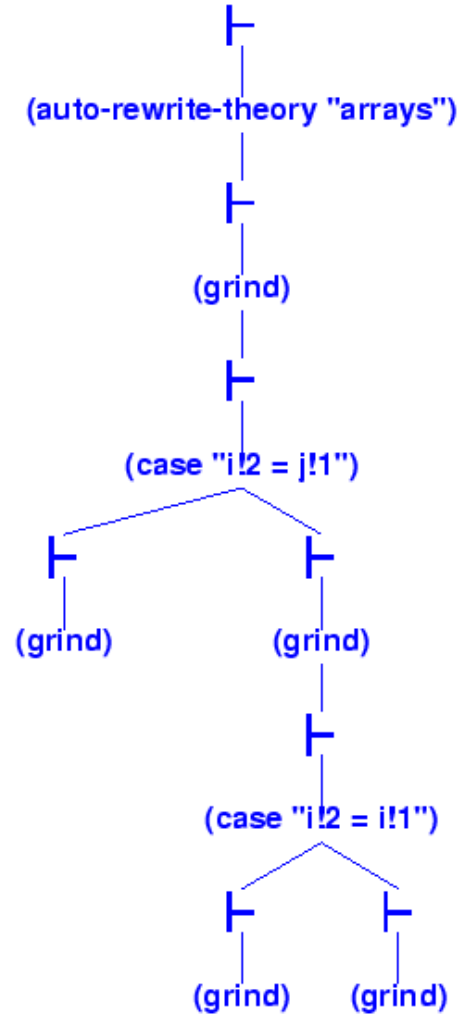


```
[2cg]: scatter
  [6zs]: assume length(put(a_0, i_0, e1_0)) = length(a_0) AND length(put(a_0, j_0, e2_0)) = length(a_0)
    [2ji]: assume length(put(put(a_0, i_0, e1_0), j_0, e2_0)) = length(a_0) AND
                  length(put(put(a_0, j_0, e2_0), i_0, e1_0)) = length(a_0)
      [6gd]: decompose
        [2yw]: instantiate put(put(a_0, j_0, e2_0), i_0, e1_0), put(put(a_0, i_0, e1_0), j_0, e2_0) in hwd
          [6ll]: goal ppq
            [2p4]: scatter
              [6ip]: case i_0 = i_1
                [2oi]: instantiate put(a_0, j_0, e2_0), i_1, e1_0 in jkt
                  [64s]: split jw2
                    [2vd]: proved (CVCL)
                    [3vd]: instantiate put(a_0, i_1, e1_0), j_0, i_1, e2_0 in gkp
                      [gin]: split 514
                        [mbq]: proved (CVCL)
                        [nbq]: proved (CVCL)
                        [obq]: instantiate a_0, i_1, e1_0 in jkt
                          [er1]: proved (CVCL)
                [3oi]: case j_0 = i_1
                  [gvb]: instantiate put(a_0, i_0, e1_0), j_0, e2_0 in jkt
                    [m5b]: split blu
                      [wtx]: proved (CVCL)
                      [xtx]: instantiate put(a_0, j_0, e2_0), i_0, j_0, e1_0 in gkp
                        [dc3]: split nr2
                          [xu6]: proved (CVCL)
                          [yu6]: proved (CVCL)
                          [zu6]: instantiate a_0, j_0, e2_0 in jkt
                            [rzj]: proved (CVCL)
```

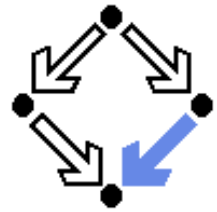
Axiomatization of Arrays (Original Contd)



```
                [hvb]: instantiate put(a_0, j_0, e2_0), i_0, i_1, e1_0 i
                  [ti2]: split lyr
                    [hgp]: proved (CVCL)
                    [igp]: proved (CVCL)
                    [jgp]: instantiate put(a_0, i_0, e1_0), j_0, i_1, e2
                      [beq]: split 36c
                        [j3l]: proved (CVCL)
                        [k3l]: proved (CVCL)
                        [l3l]: instantiate a_0, i_0, i_1, e1_0 in gkp
                          [p1h]: split srs
                            [lx5]: proved (CVCL)
                            [mx5]: proved (CVCL)
                            [nx5]: instantiate a_0, j_0, i_1, e2_0 in gk
                              [4lv]: proved (CVCL)
  [ahd]: split
    [crr]: instantiate put(a_0, i_0, e1_0), j_0, e2_0 in 4qj
      [qch]: proved (CVCL)
    [drr]: instantiate put(a_0, j_0, e2_0), i_0, e1_0 in 4qj
      [xnb]: proved (CVCL)
[3ji]: split
  [gmn]: instantiate a_0, i_0, e1_0 in 4qj
    [mfr]: proved (CVCL)
  [hmn]: instantiate a_0, j_0, e2_0 in 4qj
    [t4l]: proved (CVCL)
```

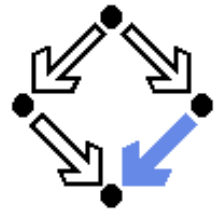


Axiomatization of Arrays (New)

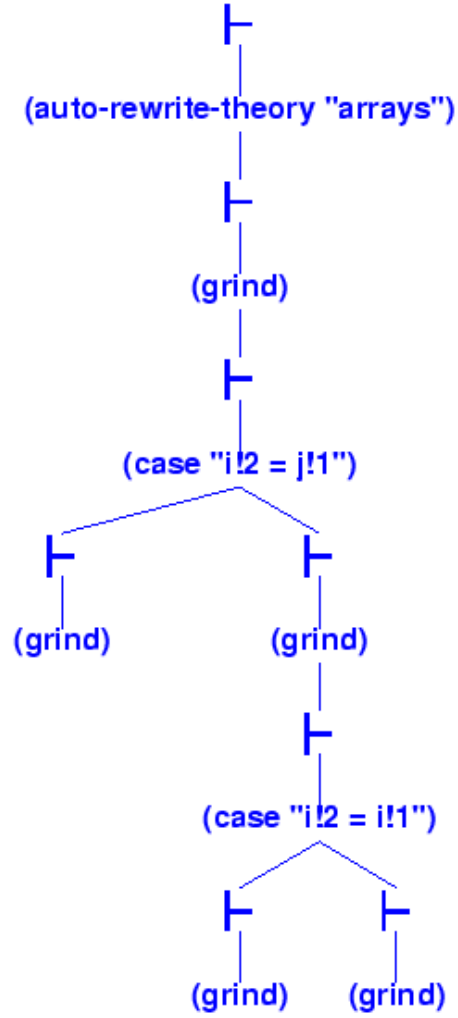


```
[2cg]: scatter
  [6zs]: instantiate put(put(a_0, i_0, e1_0), j_0, e2_0), put(put(a_0, j_0, e2_0), i_0, e1_0) in hwd
    [2ji]: goal uzo
      [6gd]: assume length(put(a_0, j_0, e2_0)) = length(a_0) AND length(put(a_0, i_0, e1_0)) = length(a_0)
        [2yw]: assume length(put(put(a_0, i_0, e1_0), j_0, e2_0)) = length(a_0) AND
                      length(put(put(a_0, j_0, e2_0), i_0, e1_0)) = length(a_0)
          [6ll]: scatter
            [2p4]: case i_1 = j_0
              [6ip]: instantiate put(a_0, j_0, e2_0), i_0, j_0, e1_0 in vv6
                [2oi]: split nr2
                  [64s]: proved (CVCL)
                  [a5s]: proved (CVCL)
                  [b5s]: instantiate put(a_0, i_0, e1_0), j_0, e2_0 in jkt
                    [jkw]: split blu
                      [bqj]: proved (CVCL)
                      [cqj]: auto jkt
                        [qyh]: proved (CVCL)
              [ajp]: case i_1 = i_0
                [ct2]: instantiate put(a_0, j_0, e2_0), i_1, e1_0 in jkt
                  [q32]: split 1i4
                    [sv3]: proved (CVCL)
                    [tv3]: instantiate put(a_0, i_1, e1_0), j_0, i_1, e2_0 in vv6
                      [hbs]: split blf
                        [ta4]: proved (CVCL)
                        [ua4]: proved (CVCL)
                        [va4]: auto jkt
                          [vay]: proved (CVCL)
```

Axiomatization of Arrays (New Contd)

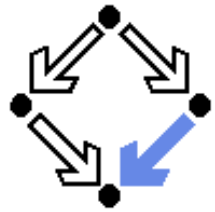


```
[dt2]: instantiate put(a_0, i_0, e1_0), j_0, i_1, e2_0 in
  [xtk]: split 36c
    [d51]: proved (CVCL)
    [e51]: proved (CVCL)
    [f51]: instantiate put(a_0, j_0, e2_0), i_0, i_1, e1_
      [fhc]: split lyr
        [fkx]: proved (CVCL)
        [gkx]: proved (CVCL)
        [hkx]: instantiate a_0, j_0, i_1, e2_0 in vv6
          [tzc]: split 25b
            [hxt]: proved (CVCL)
            [ixt]: proved (CVCL)
            [jxt]: instantiate a_0, i_0, i_1, e1_0 in vv6
              [bl2]: proved (CVCL)
[aml]: auto 4qj
  [cog]: proved (CVCL)
[3yw]: auto 4qj
  [gxf]: proved (CVCL)
```



Here moderate improvement only; automatic instantiation is no replacement for auto-rewriting.

Constructive Definition of Arrays

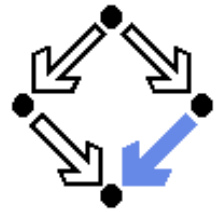


```
[adt]: expand get, length, content
  [cw2]: scatter
    [qey]: proved (CVCL)
    [rey]: assume b_0.1 = a_0.1
      [zpt]: proved (CVCL)
      [1pt]: instantiate a_0.1, b_0.1 in 1fm
        [y51]: goal 2sq
          [ku2]: scatter
            [iub]: instantiate a_0, i_0 in 3p3
              [1ef]: instantiate b_0, i_0 in 3p3
                [y4r]: instantiate i_0 in ruq
                  [kbc]: proved (CVCL)
```

```
[adt]: expand length, get, content
  [cw2]: scatter
    [qey]: proved (CVCL)
    [rey]: assume b_0.1 = a_0.1
      [zpt]: proved (CVCL)
      [1pt]: instantiate a_0.1, b_0.1 in 1fm
        [y51]: scatter
          [ku2]: auto
            [iub]: proved (CVCL)
```

Somewhat more improvements.

Linear Search: Conditions B2 and C



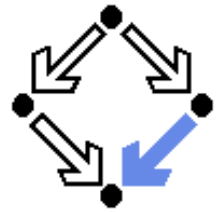
```
[q1b]: expand Invariant
  [slx]: scatter
    [a1y]: proved (Goal cqi is TRUE)
    [b1y]: instantiate j_0 in l51
      [joz]: proved (CVCL)
    [c1y]: proved (Goal cqi is TRUE)
```

```
[q1b]: expand Invariant in 6kv
  [slx]: scatter
    [a1y]: auto
      [cch]: proved (CVCL)
    [b1y]: proved (CVCL)
    [c1y]: proved (CVCL)
    [d1y]: proved (CVCL)
    [e1y]: proved (CVCL)
```

```
[dca]: expand Invariant, Output
  [tvy]: scatter
    [dcu]: instantiate j_0 in l51
      [t4c]: proved (CVCL)
    [ecu]: goal ueq
      [kel]: scatter
        [ulf]: proved (CVCL)
        [vlf]: instantiate j_0 in l51
          [rzb]: proved (CVCL)
    [fcu]: instantiate j_0 in l51
      [blt]: proved (CVCL)
    [gcu]: proved (CVCL)
```

```
[dca]: expand Invariant, Output in zfg
  [tvy]: scatter
    [dcu]: auto
      [t4c]: proved (CVCL)
    [ecu]: split pkg
      [kel]: proved (CVCL)
      [lel]: scatter
        [lvn]: auto
          [lap]: proved (CVCL)
    [fcu]: auto
       [blt]: proved (CVCL)
    [gcu]: proved (CVCL)
```

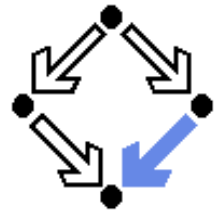
Binary Search: Lemma L and Condition A



```
[mca]: flatten
  [c3f]: induction j z42
    [3da]: scatter
      [s3k]: proved (CVCL)
    [4da]: scatter
      [jdt]: instantiate k_0 in cgs
        [4kl]: instantiate j_0 in iuh
          [ji1]: proved (CVCL)

[bca]: expand Input, Invariant in ne5
  [fuo]: scatter
    [bxg]: simplify
      [ft4]: proved (Goal cqi is TRUE)
    [cxg]: proved (Goal cqi is TRUE)
    [dxg]: proved (Goal cqi is TRUE)
    [exg]: simplify
      [khw]: proved (Goal cqi is TRUE)
```

```
[mca]: flatten
  [c3f]: induction j z42
    [3da]: scatter
      [s3k]: proved (CVCL)
    [4da]: scatter
      [jdt]: auto
        [4kl]: proved (CVCL)

[bca]: expand Input, Invariant in ne5
  [fuo]: scatter
    [bxg]: proved (CVCL)
    [cxg]: typeaxiom a in length
      [3zf]: proved (CVCL)
    [dxg]: proved (CVCL)
    [exg]: auto
      [khw]: proved (CVCL)
```

Binary Search: Condition B1 and C



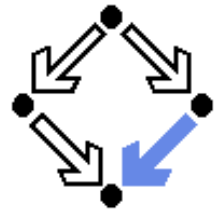
```
[p1b]: expand Invariant
  [lf6]: decompose
    [p6i]: instantiate low/2+high/2 in 6hu
      [l3h]: scatter
        [p3f]: proved (Goal cqi is TRUE)
        [q3f]: proved (Goal cqi is TRUE)
        [r3f]: proved (Goal cqi is TRUE)
        [s3f]: instantiate 0 in unk
          [asu]: proved (CVCL)
        [t3f]: proved (Goal cqi is TRUE)
      [m3h]: proved (CVCL)
```

```
[p1b]: instantiate low/2+high/2 in 6hu
  [lf6]: expand Invariant
    [p6i]: scatter
      [l3h]: proved (CVCL)
      [m3h]: proved (CVCL)
      [n3h]: proved (CVCL)
      [o3h]: auto
        [e1h]: proved (CVCL)
      [p3h]: proved (CVCL)
  [mf6]: proved (CVCL)
```

```
[dca]: expand Invariant, Output
  [tvy]: scatter
    [dcu]: instantiate j_0 in zvo
      [t4c]: instantiate j_0 in puc
        [dcu]: proved (CVCL)
    [ecu]: goal ueq
      [kel]: scatter
        [ulf]: proved (Goal cqi is TRUE)
        [vlf]: instantiate j_0 in zvo
          [rzb]: instantiate j_0 in puc
            [vor]: proved (CVCL)
```

```
[dca]: expand Invariant, Output in azp
  [tvy]: scatter
    [dcu]: auto
      [t4c]: proved (CVCL)
    [ecu]: split pkg
      [kel]: proved (CVCL)
      [lel]: scatter
        [lvn]: auto
          [lap]: proved (CVCL)
```

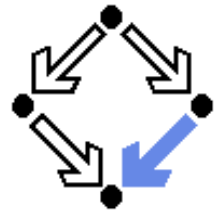
Binary Search: Condition B2 (similar B3)



```
[q1b]: instantiate low/2+high/2 in 6hu
  [slx]: expand Invariant
    [a1y]: scatter
      [cch]: proved (Goal cqi is TRUE)
      [dch]: instantiate 0 in unk
        [xmk]: proved (CVCL)
      [ech]: proved (Goal cqi is TRUE)
      [fch]: proved (Goal cqi is TRUE)
      [gch]: instantiate j_0 in puc
        [mkm]: split ixd
          [wgk]: proved (CVCL)
          [xgk]: lemma L
            [d4y]: instantiate
                floor(low/2+high/2), j_0 in z42
              [xtj]: proved (CVCL)
              [ytj]: proved (CVCL)
          [ygk]: proved (CVCL)
      [hch]: proved (Goal cqi is TRUE)
      [ich]: instantiate j_0 in zvo
        [1iy]: proved (CVCL)
  [tlx]: proved (CVCL)
```

```
[q1b]: instantiate low/2+high/2 in 6hu
  [slx]: expand Invariant
    [a1y]: scatter
      [cch]: proved (CVCL)
        [dch]: lemma L
          [xmk]: auto z42
            [d45]: proved (CVCL)
        [ech]: auto
          [5la]: proved (CVCL)
        [fch]: proved (CVCL)
        [gch]: proved (CVCL)
        [hch]: proved (CVCL)
  [tlx]: proved (CVCL)
```

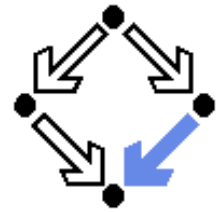
Client/Server System: Formal Model



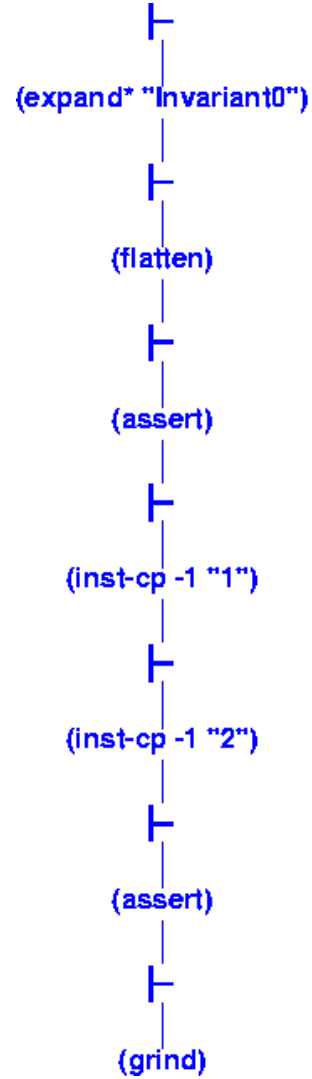
```
Index: TYPE =
  SUBTYPE(LAMBDA(x:INT):
    x=1 OR x=2);
Index0: TYPE =
  SUBTYPE(LAMBDA(x:INT):
    x=0 OR x=1 OR x=2);
PCBASE: TYPE;
R: PCBASE;
S: PCBASE;
C: PCBASE;
PC: TYPE =
  SUBTYPE(LAMBDA(x:PCBASE):
    x=R OR x=S OR x=C);
PCs: AXIOM
  R /= S AND R /= C AND S /= C;
pc: Index->PC;
pc0: Index->PC;
request: Index->BOOLEAN;
request0: Index->BOOLEAN;
answer: Index->BOOLEAN;
answer0: Index->BOOLEAN;
given: Index0;
given0: Index0;
waiting: Index0;
waiting0: Index0;
sender: Index0;
sender0: Index0;
rbuffer: Index -> BOOLEAN;
rbuffer0: Index -> BOOLEAN;
sbuffer: Index -> BOOLEAN;
sbuffer0: Index -> BOOLEAN;
```

```
IC: (PC, BOOLEAN, BOOLEAN) -> BOOLEAN = ...
IS: (Index0, Index0, Index0, Index->BOOLEAN, Index->BOOLEAN) -> BOOLEAN = ...
Initial: BOOLEAN = ...

RC: (PC, BOOLEAN, BOOLEAN, PC, BOOLEAN, BOOLEAN)->BOOLEAN = ...
RS: (Index0, Index0, Index0, Index->BOOLEAN, Index->BOOLEAN,
     Index0, Index0, Index0, Index->BOOLEAN, Index->BOOLEAN)->BOOLEAN = ...
External: (Index, PC, BOOLEAN, BOOLEAN, PC, BOOLEAN, BOOLEAN,
           Index0, Index0, Index0, Index->BOOLEAN, Index->BOOLEAN,
           Index0, Index0, Index0, Index->BOOLEAN, Index->BOOLEAN)->BOOLEAN =
   ...
Next: BOOLEAN = ... % disjunction of 10 cases

Invariant: (Index->PC, Index->BOOLEAN, Index->BOOLEAN,
            Index0, Index0, Index0, Index->BOOLEAN, Index->BOOLEAN) -> BOOLEAN
            = ...

Inv1: FORMULA
  Initial =>
    Invariant(pc, request, answer, given, waiting, sender, rbuffer, sbuffer);
Inv2: FORMULA
  Invariant(pc, request, answer, given, waiting, sender,
    rbuffer, sbuffer) AND Next =>
  Invariant(pc0, request0, answer0, given0, waiting0, sender0,
    rbuffer0, sbuffer0);
MutEx: FORMULA
  Invariant(pc, request, answer, given, waiting, sender, rbuffer, sbuffer) =>
  NOT(pc(1) = C AND pc(2) = C);
```

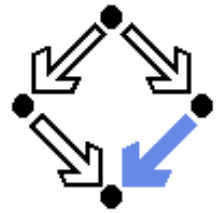
Client/Server System: Condition MutEx



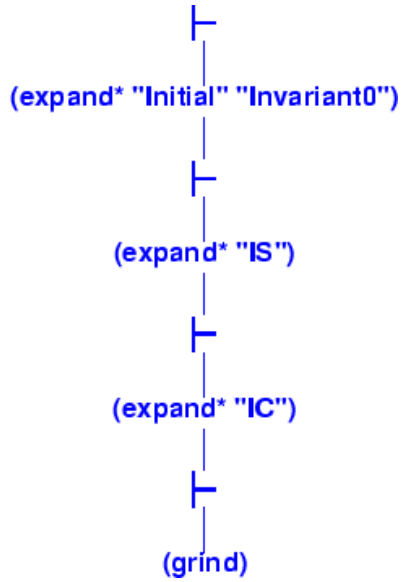
```
[z3f]: expand Invariant, IC, IS
  [nhn]: scatter
    [znj]: auto
      [n1u]: proved (CVCL)
```



Client/Server System: Condition Inv1

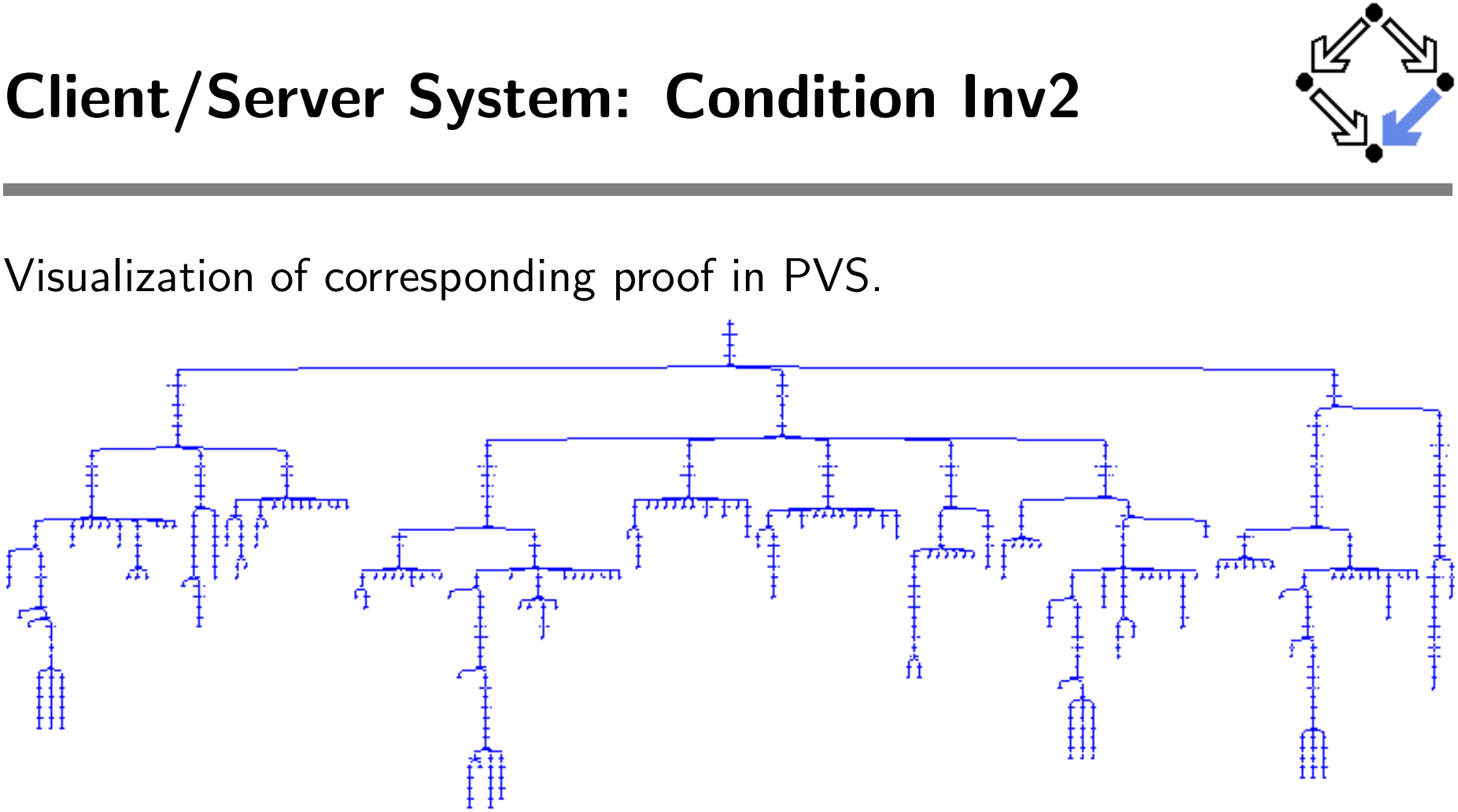


```
[oas]: expand Initial, Invariant, IC, IS
  [eij]: scatter
    [5ul]: auto
      [uvj]: proved (CVCL)
    [6ul]: auto
      [2u6]: proved (CVCL)
    [avl]: auto
      [cuv]: proved (CVCL)
    [bvl]: auto
      [jtl]: proved (CVCL)
    [cvl]: auto
      [qsb]: proved (CVCL)
    [dvl]: auto
      [xrx]: proved (CVCL)
    [evl]: auto
      [5qn]: proved (CVCL)
    [fvl]: auto
      [fqd]: proved (CVCL)
    [gvl]: auto
      [mpz]: proved (CVCL)
    [hvl]: proved (CVCL)
    [h5h]: auto
      [p3z]: proved (CVCL)
    [i5h]: auto
      [gjb]: proved (CVCL)
    [j5h]: auto
      [4vi]: proved (CVCL)
    [k5h]: auto
      [ucq]: proved (CVCL)
    [l5h]: auto
      [1px]: proved (CVCL)
    [m5h]: proved (CVCL)
    [n5h]: proved (CVCL)
    [o5h]: proved (CVCL)
    [p5h]: proved (CVCL)
    [q5h]: proved (CVCL)
    [q5i]: proved (CVCL)
    [r5i]: proved (CVCL)
    [s5i]: proved (CVCL)
    [t5i]: proved (CVCL)
    [u5i]: auto
      [1br]: proved (CVCL)
    [v5i]: auto
      [roy]: proved (CVCL)
    [w5i]: auto
      [i26]: proved (CVCL)
    [x5i]: proved (CVCL)
    [y5i]: auto
      [wuo]: proved (CVCL)
    [z5i]: auto
      [nbw]: proved (CVCL)
    [z5j]: auto
      [nbn]: proved (CVCL)
    [15j]: auto
      [eou]: proved (CVCL)
    [25j]: proved (CVCL)
      [35j]: proved (CVCL)
    [45j]: proved (CVCL)
    [55j]: proved (CVCL)
    [65j]: proved (CVCL)
```



Single application of autostar.

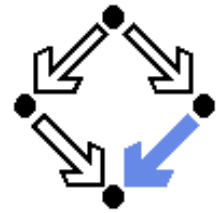
Client/Server System: Condition Inv2



Visualization of corresponding proof in PVS.

Ten main branches requiring dozens of (partially creative) steps each.

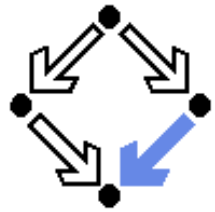
Client/Server System: Condition Inv2



```
[pas]: scatter
  [lbh]: expand Next
    [pzi]: split bfv
      [leh]: decompose
        [pkr]: expand RS
          [lpn]: split 5xv
            [pt6]: expand Invariant
              [lcw]: scatter
                [puh]: auto
                  [l43]: proved (CVCL)
                ... (20 times)
                [tuh]: proved (CVCL)
                ... (15 times)
            [qt6]: expand Invariant
              [snq]: scatter
                [avi]: auto
                  [cct]: proved (CVCL)
                ... (26 times)
                [gvi]: proved (CVCL)
                ... (6 times)
            [rt6]: scatter
              [zyk]: expand Invariant
                [rvj]: scatter
                  [zgj]: auto
                    [rhd]: proved (CVCL)
                  ... (31 times)
                  [2f3]: proved (CVCL)
                  ... (1 times)
[st6]: scatter
  [aef]: expand Invariant
    [cwk]: scatter
      [ql6]: auto
        [seg]: proved (CVCL)
      ... (21 times)
      [wl6]: proved (CVCL)
      ... (12 times)
[tt6]: scatter
  [hp6]: expand Invariant
    [twl]: scatter
      [hqv]: auto
        [tbj]: proved (CVCL)
      ... (27 times)
      [nqv]: proved (CVCL)
      ... (6 times)
[meh]: scatter
  [w3z]: expand External
    [3rk]: split lhe
      [g4b]: scatter
        [mdh]: expand Invariant
          [wzf]: scatter
            [3ys]: auto
              [gsh]: proved (CVCL)
            ... (36 times)
[h4b]: scatter
  [tob]: expand Invariant
    [h1g]: scatter
      [t4i]: auto
        [hpk]: proved (CVCL)
      ... (36 times)
[neh]: scatter
  [4oc]: expand RC
    [nuh]: split nwz
      [4ge]: scatter
        [ney]: expand Invariant
          [45d]: scatter
            [nui]: auto
              [4wr]: proved (CVCL)
            ... (36 times)
          [5ge]: scatter
            [ups]: expand Invariant
              [o6e]: scatter
                [ez5]: auto
                  [5tu]: proved (CVCL)
                ... (36 times)
          [6ge]: scatter
            [21m]: expand Invariant
              [66f]: scatter
                [24u]: auto
                  [6qx]: proved (CVCL)
                ... (36 times)
```

Ten main branches requiring only single application of `autostar` each.

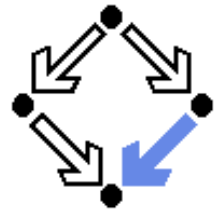
Outlook



- User and implementation documentation.
  - Release version 1.0 in this summer.
- Generic interface for decision procedures.
  - CVCL 2.5 is more restricted than CVCL 2.0 (first order only).
  - Participants in SMT-LIB competition are promising candidates.
- Component for rewriting proofs.
  - Use implications as rewrite rules.
- Improved formula/term represention.
  - More abstract and easier to use than current one.

Potential for programming projects and diploma theses.

Acknowledgements



The ProofNavigator is based on various pieces of open source software.

- CVC Lite 2.0
- RIACA OpenMath Library 2.0
- General Purpose Hash Function Algorithms Library
- ANTLR 2.7.6b2
- Eclipse Standard Widget Toolkit 3.1.2
- Mozilla Browser 1.7.8 GTK2
- GIMP Toolkit GTK+ 2.X
- Java Development Kit 5.0
- Tango Icon Library

It is important to contribute to the open source pool.