Improving the WCET computation time by IPET using CFG partitioning

C. BALLABRIGA, H. CASSÉ

{ballabri, casse}@irit.fr

TRACES – IRIT - Université de Toulouse - France

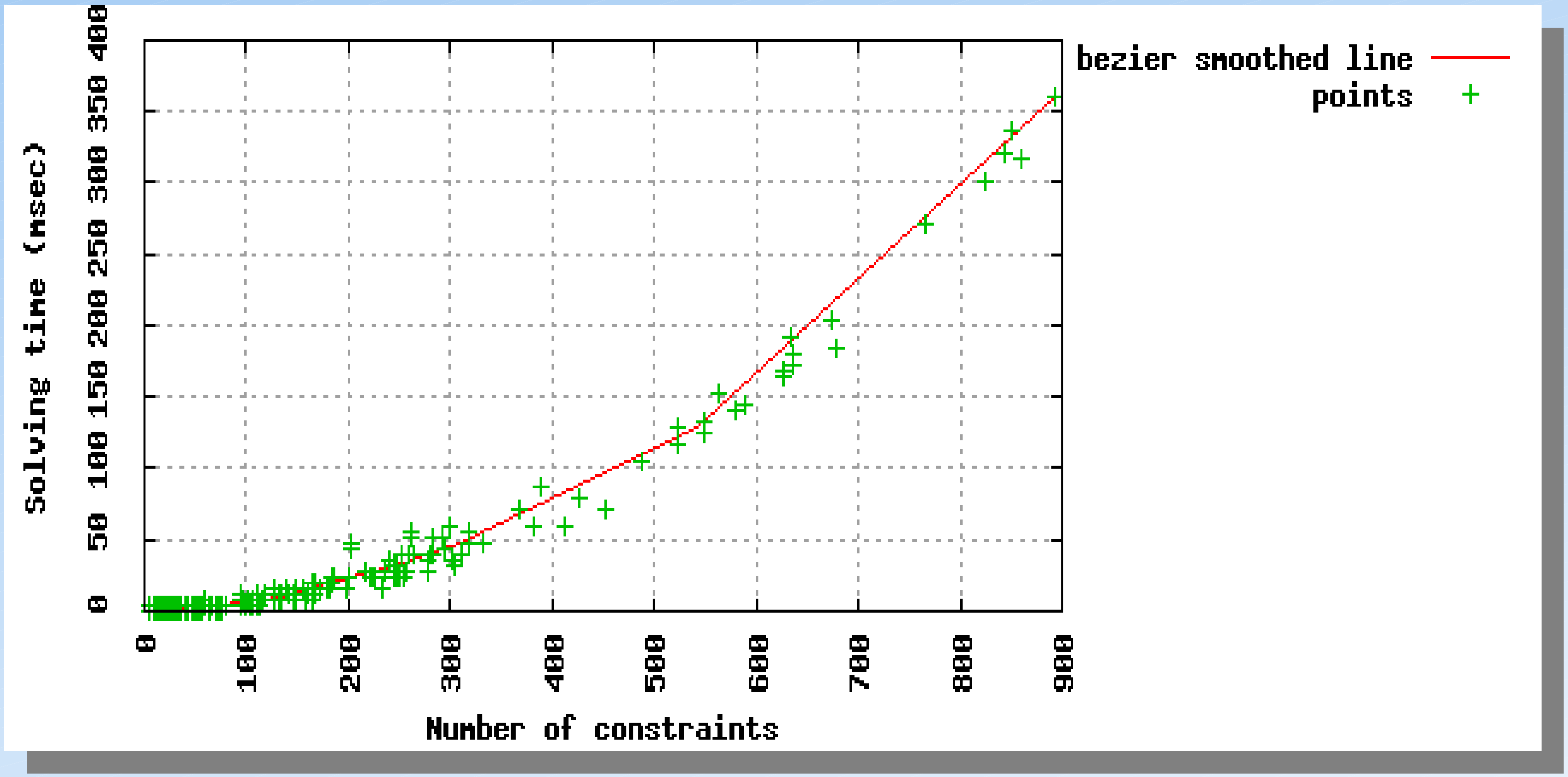
WCET Computation

- WCET computation by static analysis
  - program control flow analysis
  - architecture effects analysis
  - WCET computation → IPET

- IPET : widely-used WCET computation approach
  - express program flow and hardware effects using an ILP system
  - an ILP solver is used to compute the WCET
    (an objective function to maximize)

ILP solving

- ILP solving time is high, and increases non-linearly with system size

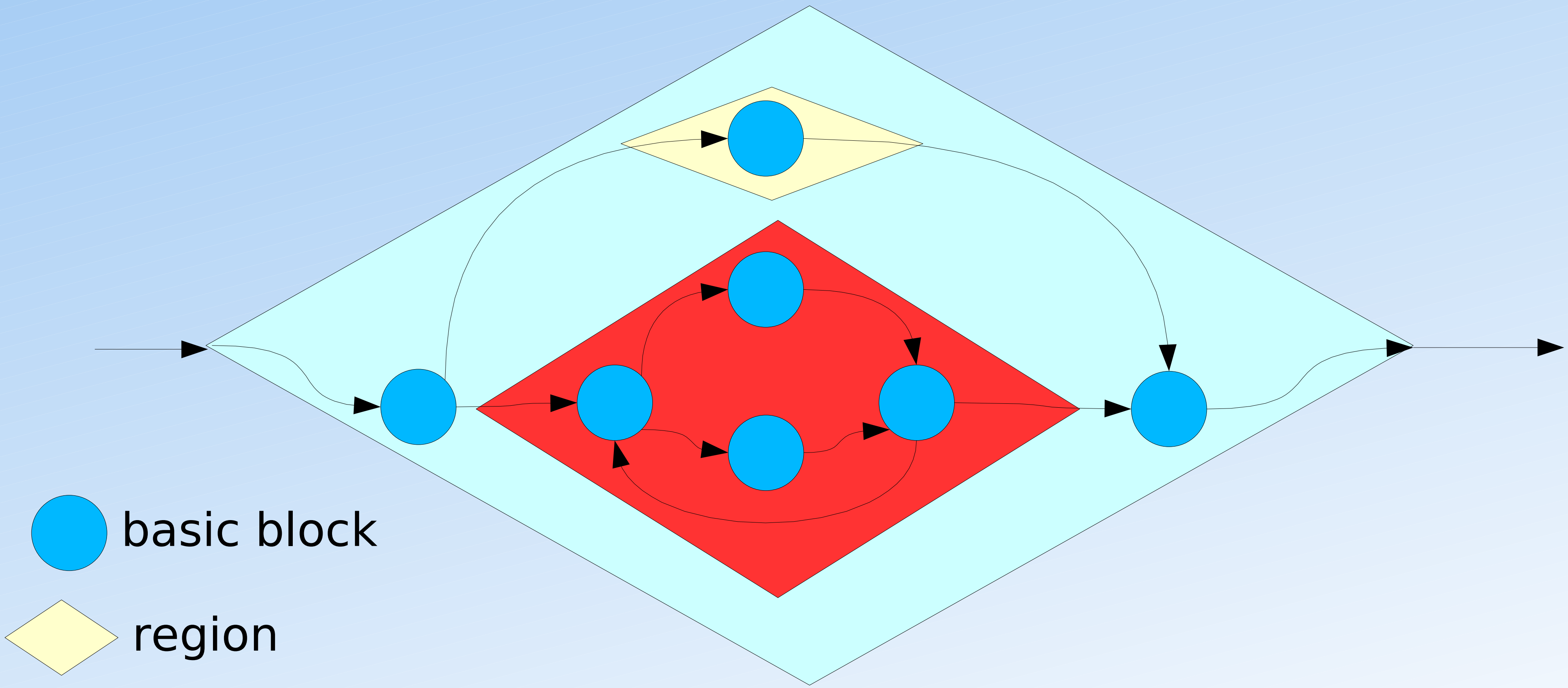


*(ILP solving time, with **lp_solve**)*

- solution: split systems into smaller subsystems

regions : basic idea

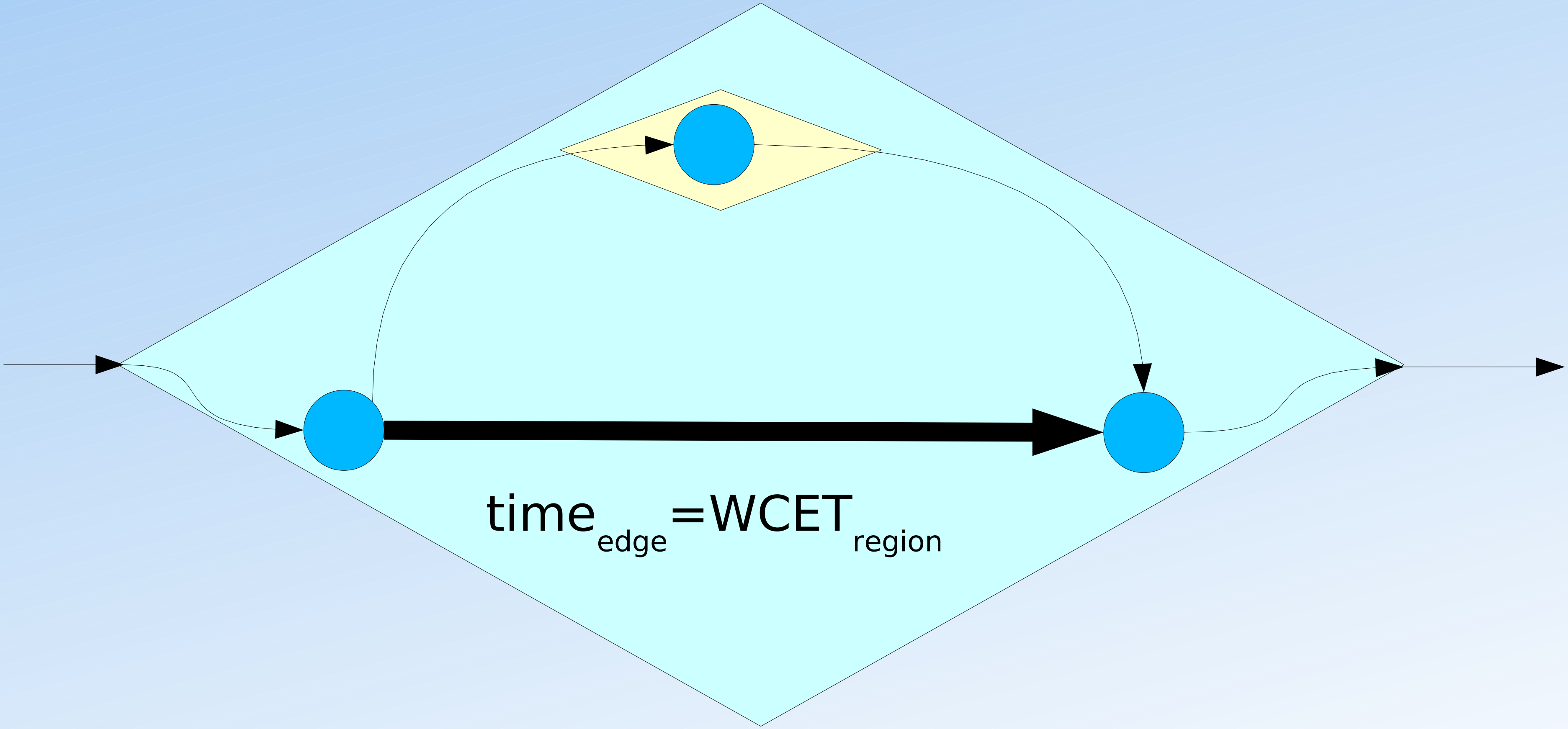
- single-entry, single exit regions (SESE)
- WCET of a region computable independently



(R. Johnson et al. - The Program Structure Tree: Computing Control Regions in Linear Time – SIGPLAN Conference on Programming Language Design and Implementation, 1994)

regions : basic idea

- single-entry, single exit regions
- WCET of a region computable independently
- computed WCET used in parent region

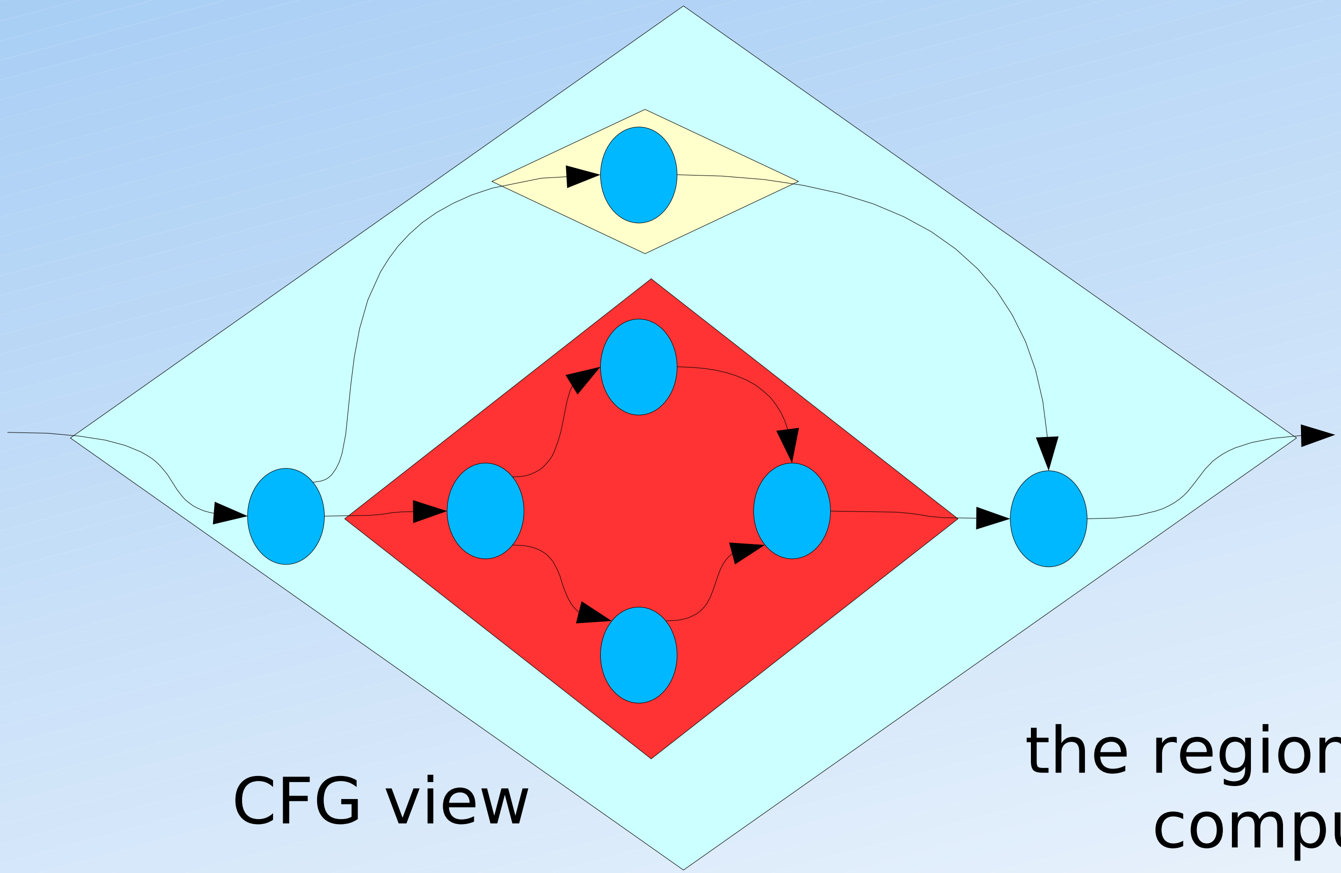


$\text{time}_{\text{edge}} = \text{WCET}_{\text{region}}$

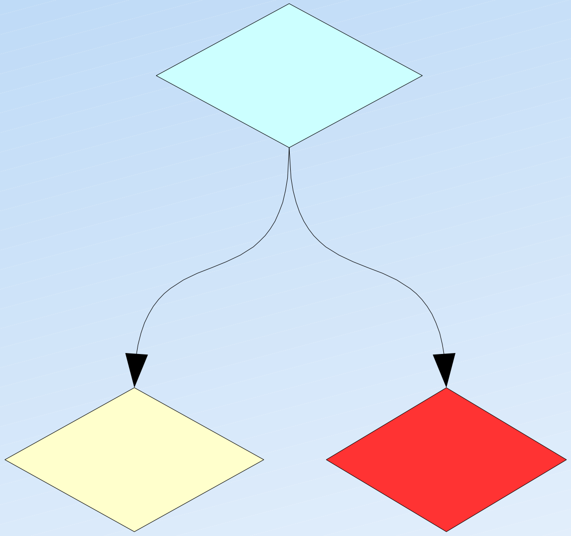
(region now modeled by a single edge)

Program Structure Tree (PST)

- regions can be structured into a tree
- compute the WCET by a bottom-up visit



CFG view



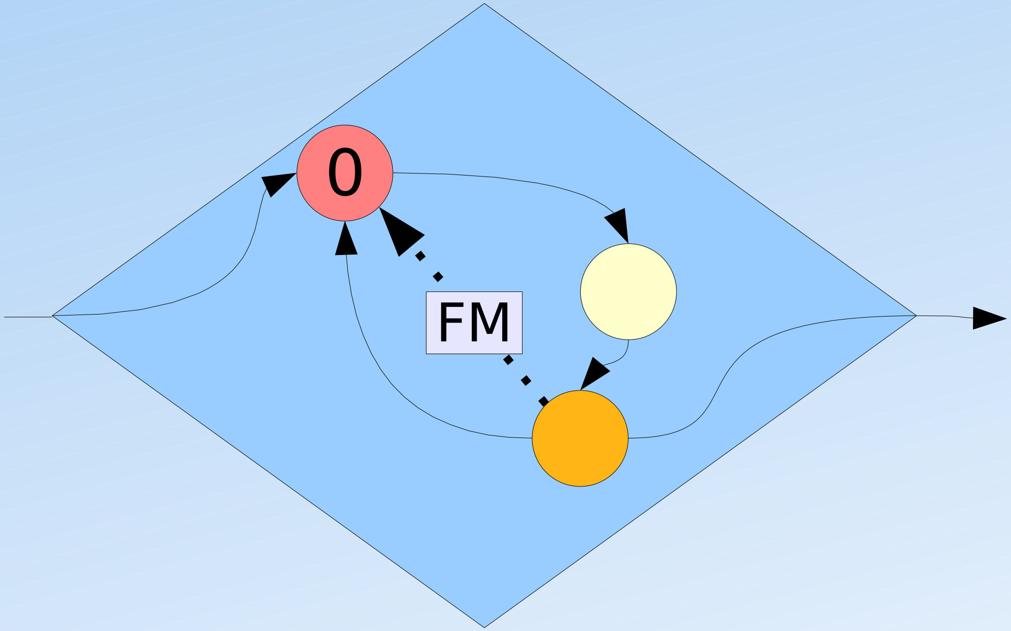
PST view

the regions in the PST must be computable independently without altering the final WCET

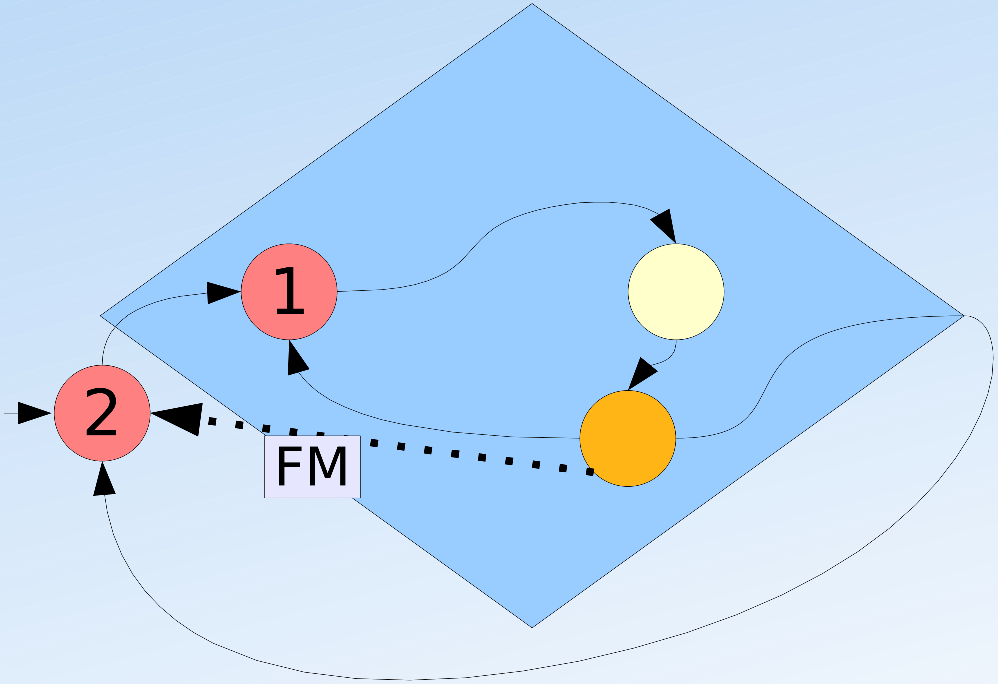
Cache issues

- main problem: Persistence
- parametrized persistence is easier to handle

  (C. Ballabriga, H. Cassé – Improving the First-Miss Computation in Set-Associative Instruction Caches – ECRTS'08)



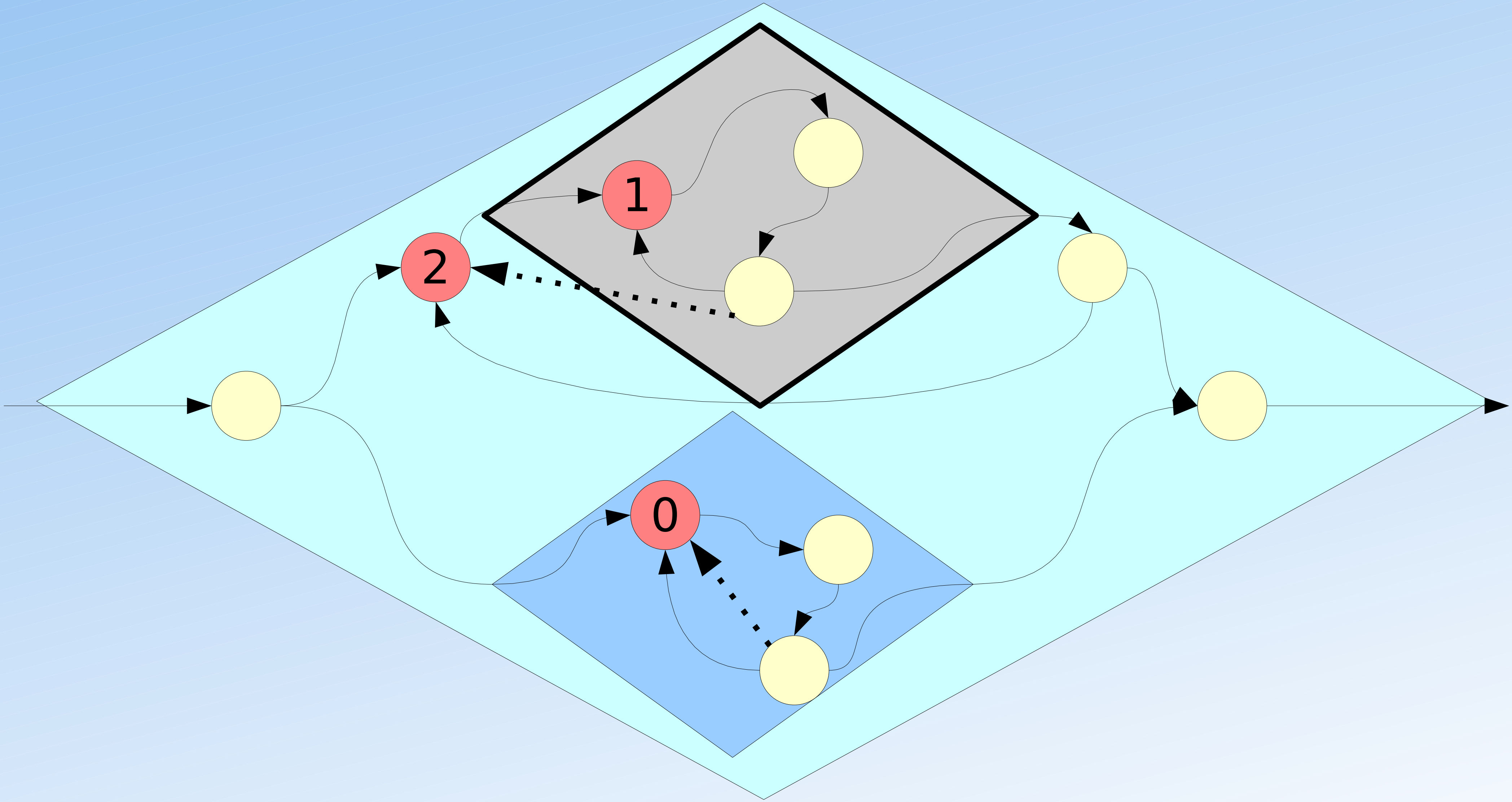
✔ OK

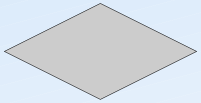


✗ NOT OK

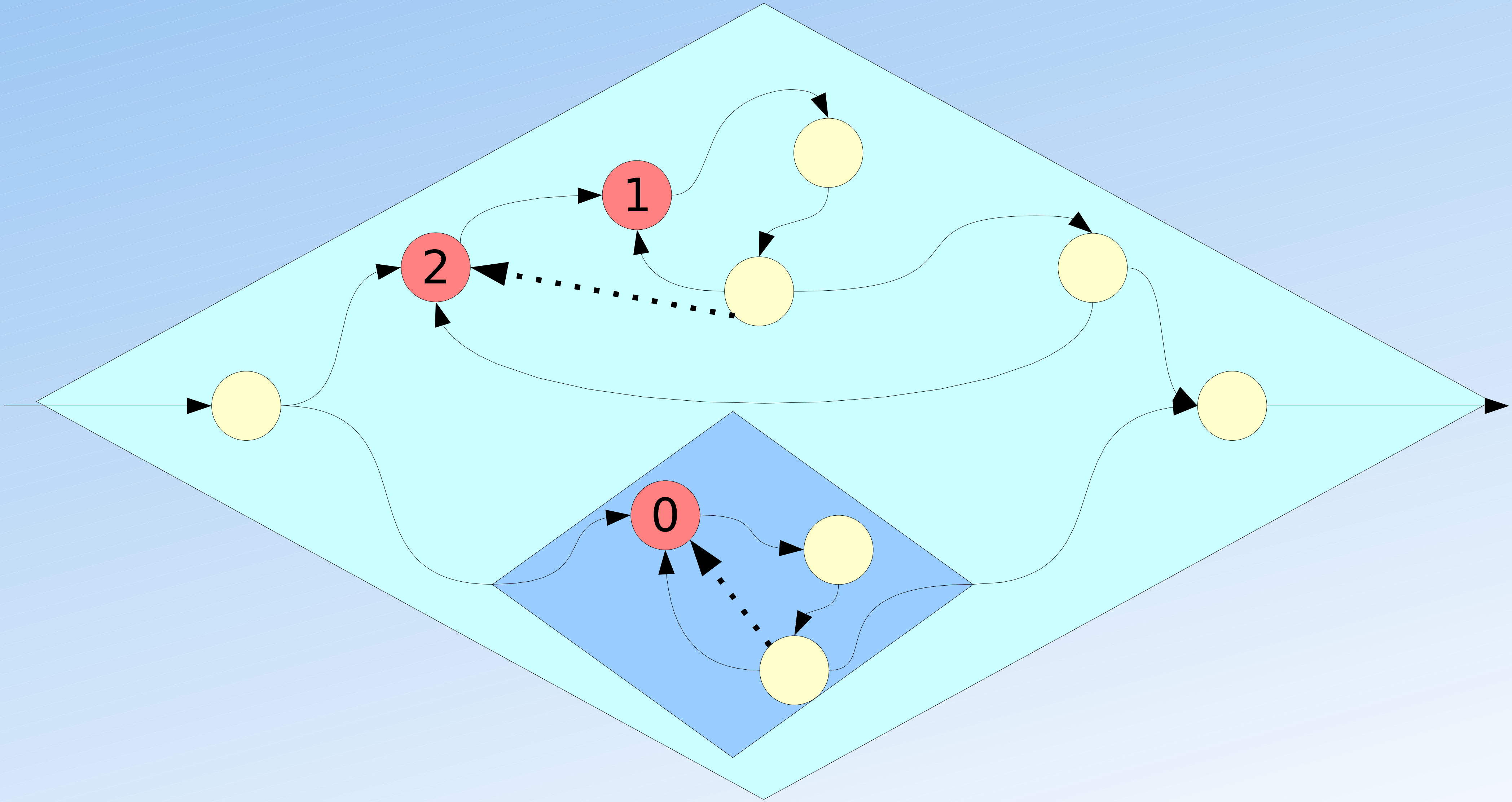
because WCET depends on block 2

Eliminating infeasible regions



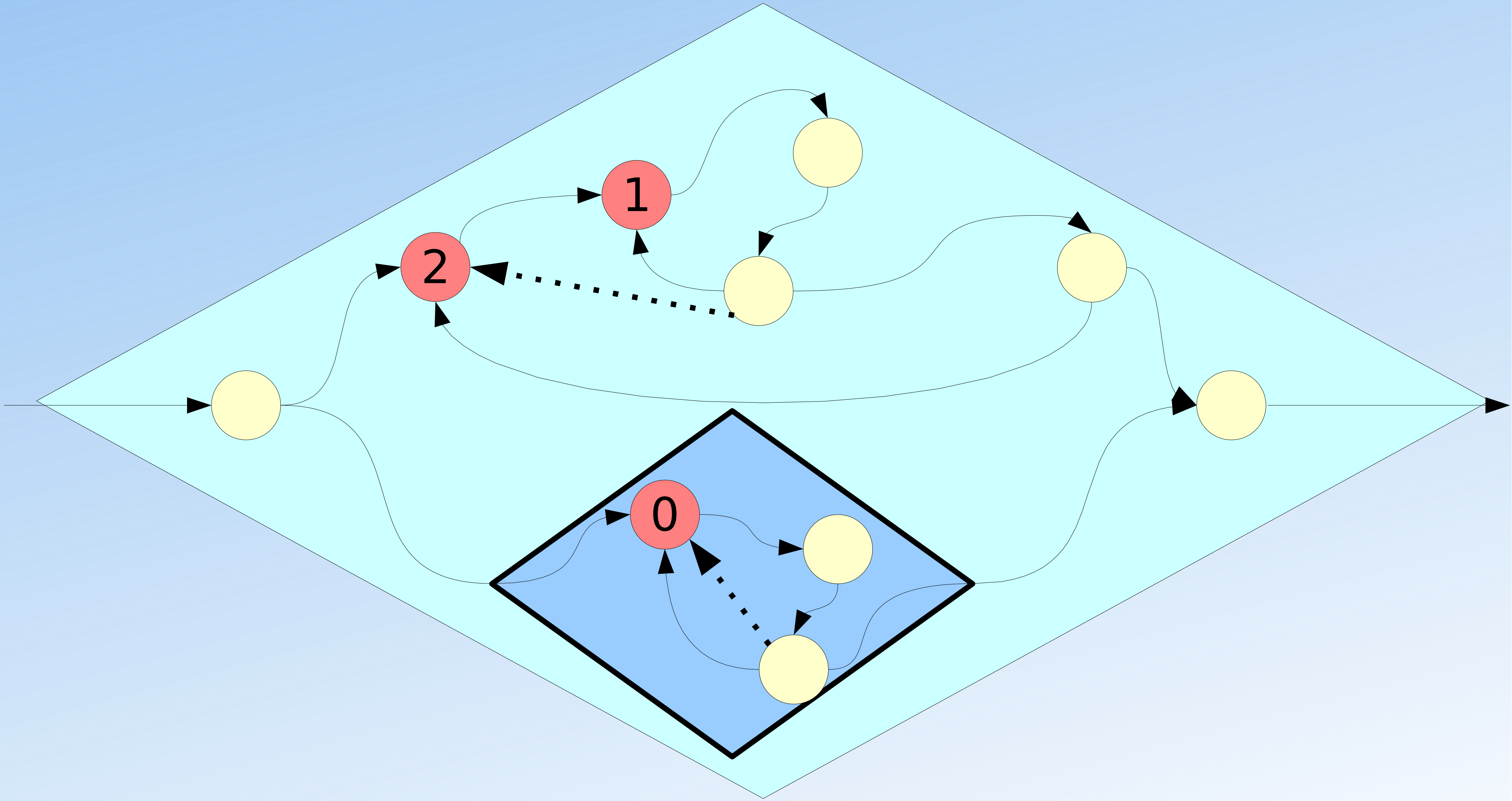

infeasible region

Eliminating infeasible regions



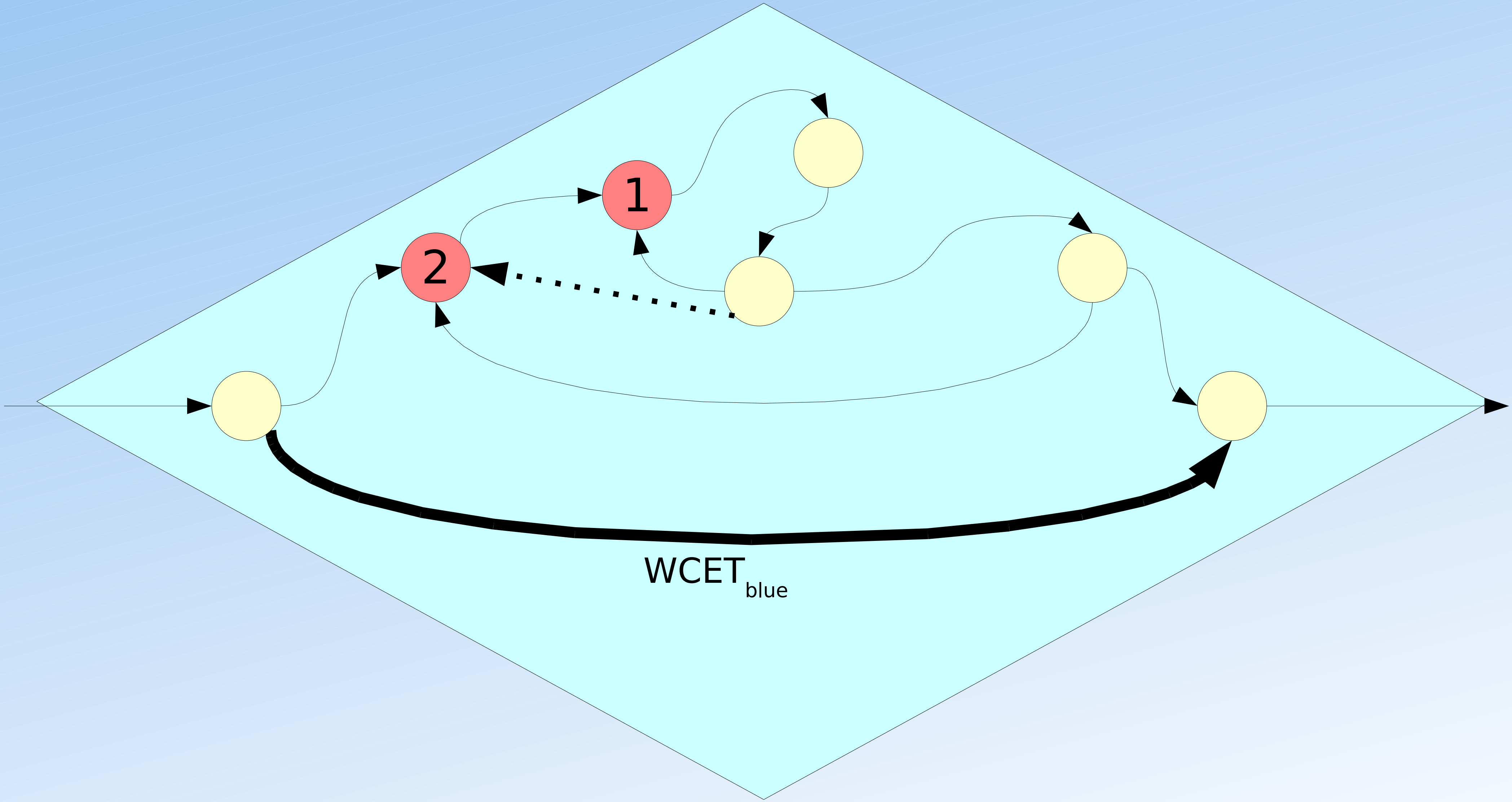
◆ infeasible region

Eliminating infeasible regions



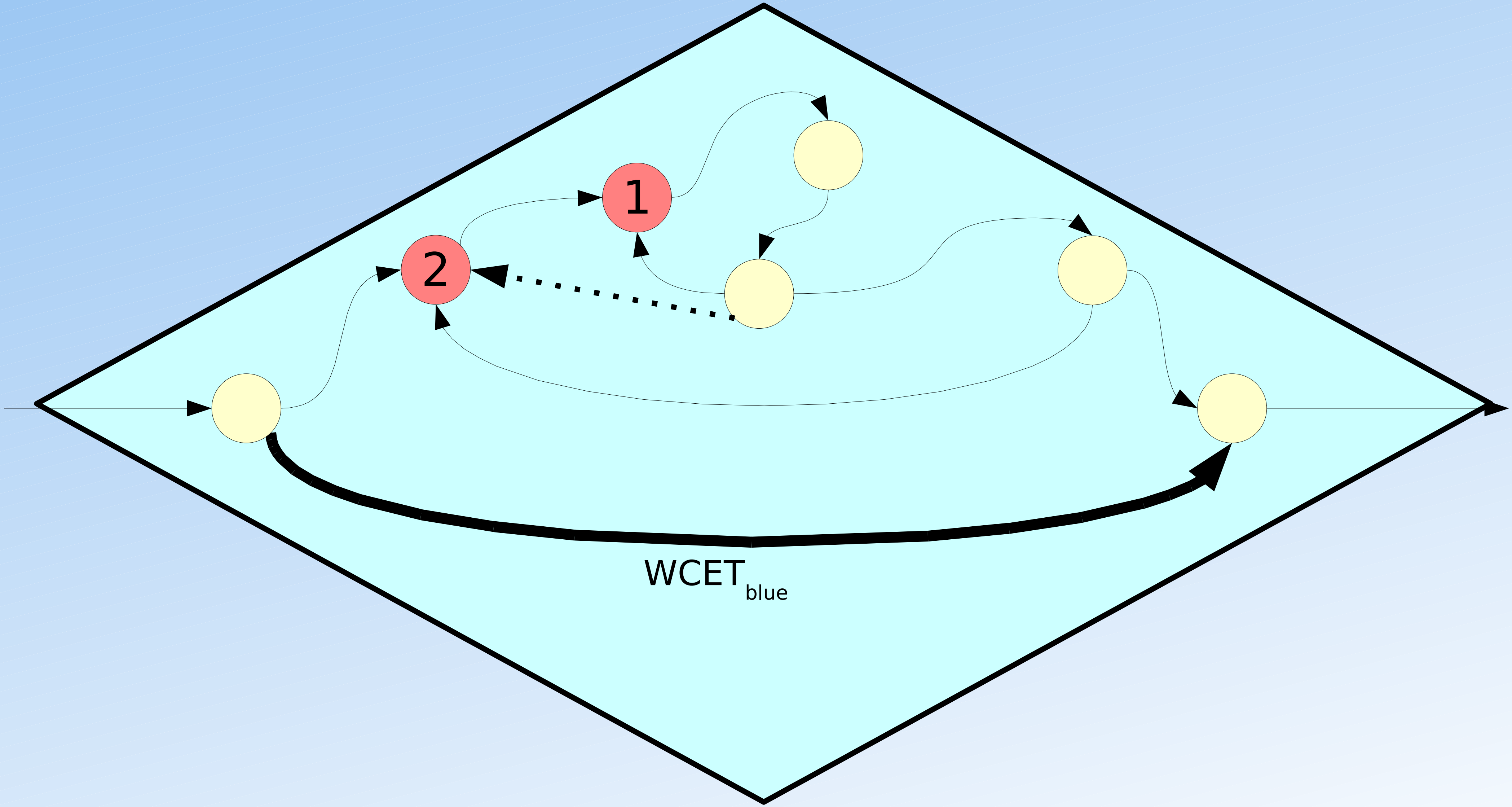
compute other regions

Eliminating infeasible regions



compute other regions

Eliminating infeasible regions



compute other regions

Eliminating infeasible regions

compute other regions

Pipeline: Exegraph

- Exegraph: compute BB execution time using a graph and BB predecessors

(C. Rochange, P. Sainrat - A Context-Parameterized Model for Static Analysis of Execution Times – HiPEAC'2007)

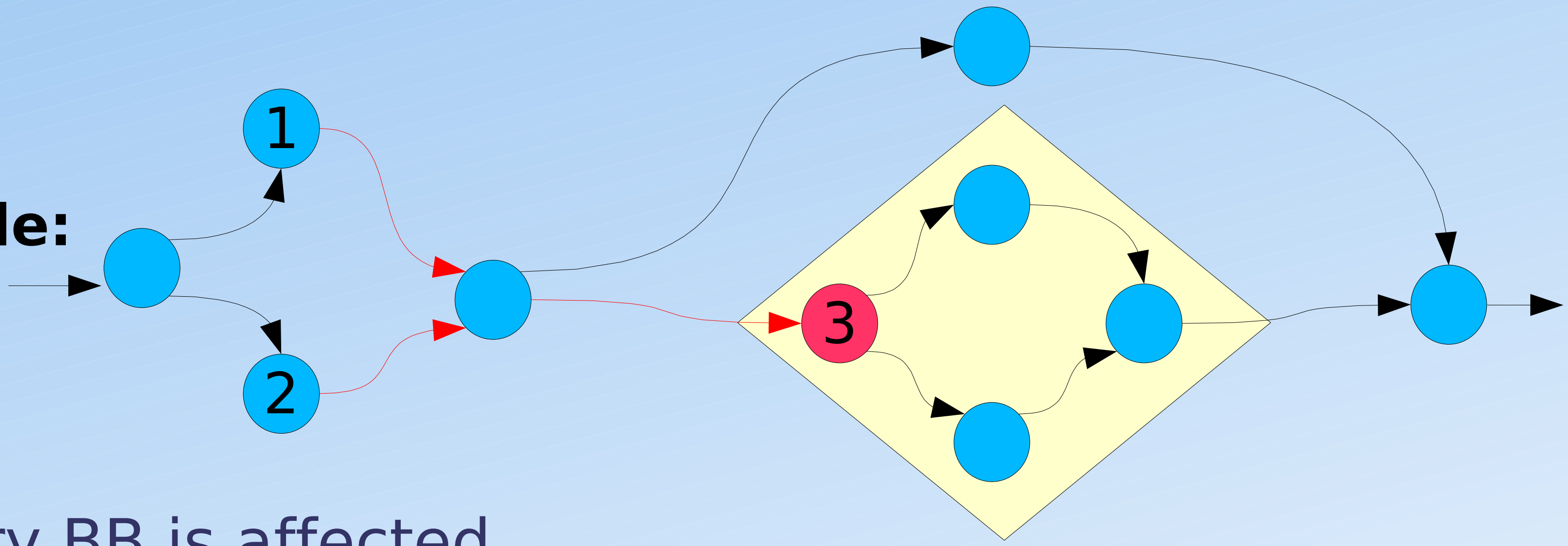
- several context-handling modes
  1. one time whatever the predecessors
  2. one time for each direct predecessor
  3. one time for each sequence of 2 pred.
  4. one time for each sequence of more than 2 pred.

in cases 1 and 2 Exegraph does not introduce region dependencies

Pipeline: Exegraph

sequences of two basic blocks

example:

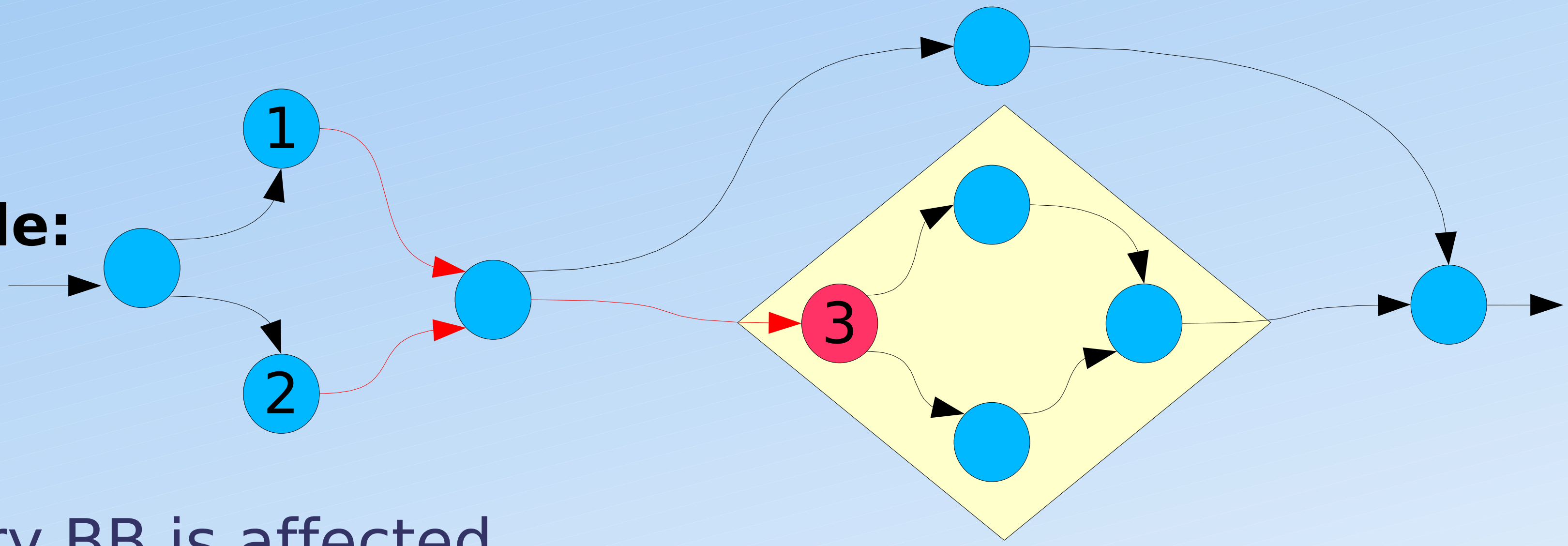


- only entry BB is affected
- all paths go through this BB
- we can compute the region WCET minus this BB.

Pipeline: Exegraph

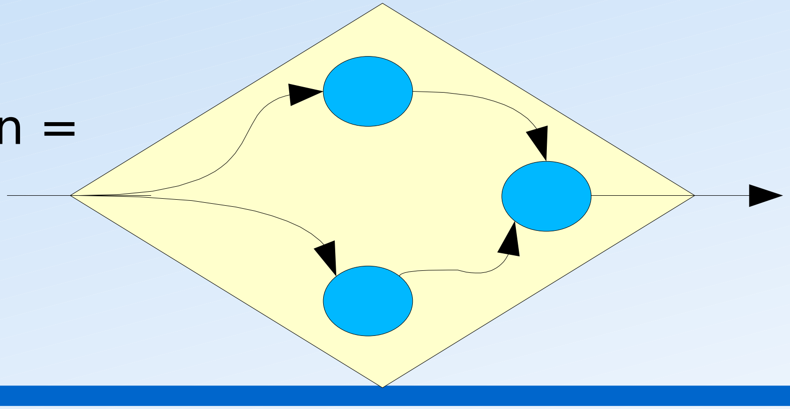
sequences of two basic blocks

example:

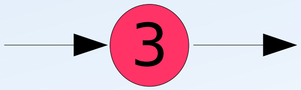


- only entry BB is affected
- all paths go through this BB
- we can compute the region WCET minus this BB.

common =

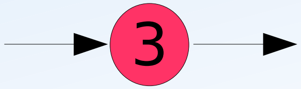


delta1 =



with seq. 1

delta2 =

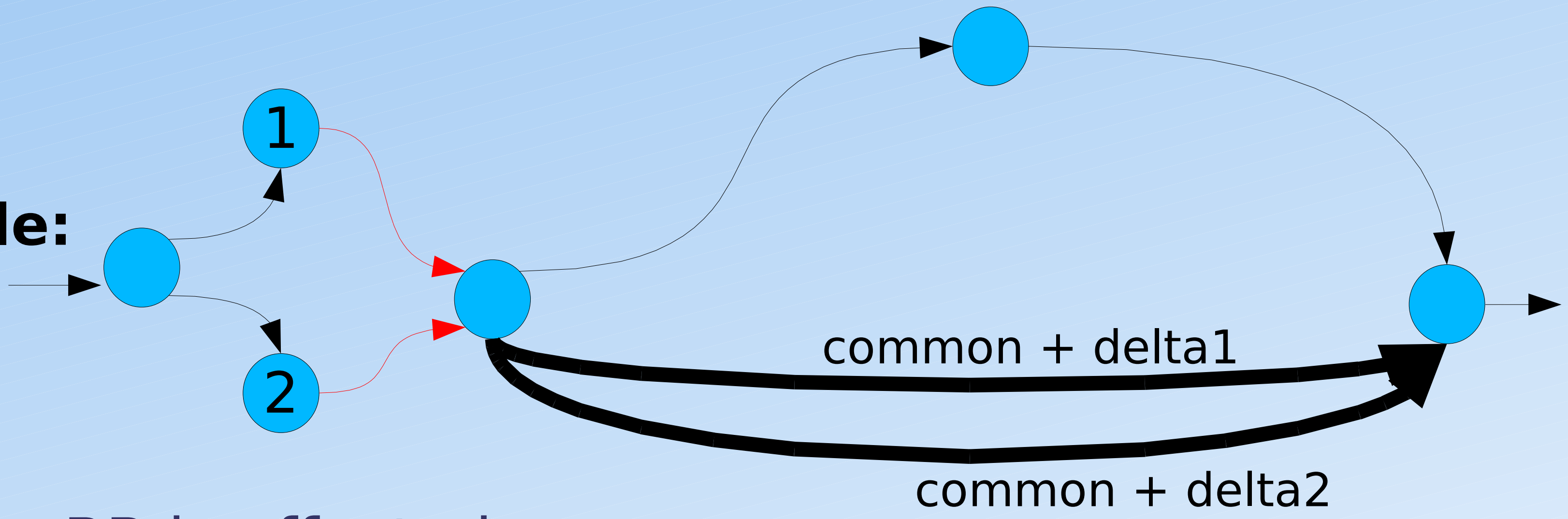


with seq. 2

Pipeline: Exegraph

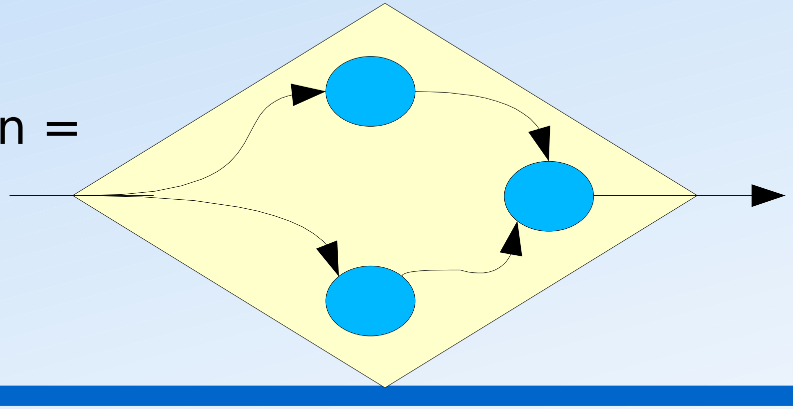
sequences of two basic blocks

example:

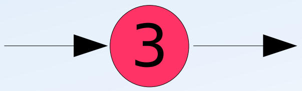


- only entry BB is affected
- all paths go through this BB
- we can compute the region WCET minus this BB.

common =

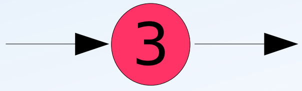


delta1 =



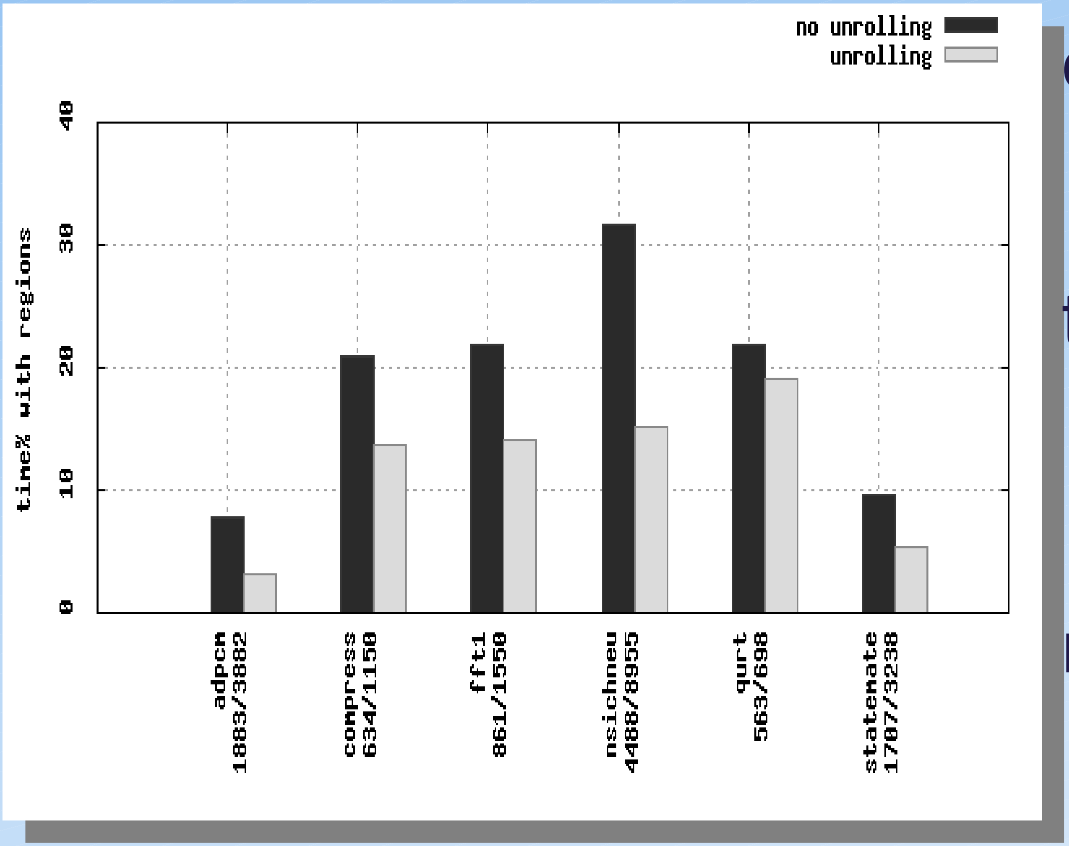
with seq. 1

delta2 =



with seq. 2

Conclusion



experimentation software

- OTAWA, our WCET computation tool

target architecture

- simple pipeline (Exegraph)
- 4-way associative instruction cache, LRU policy (categories)

results

- on average, 6.5 times faster

future works:

- test with others solvers
- apply to COTS
- check adaptability to others hardware analyses