Making Dynamic Memory Allocation Static To Support WCET Analyses

Jörg Herter Jan Reineke

Department of Computer Science
Saarland University

WCET Workshop, June 2009

Current Situation

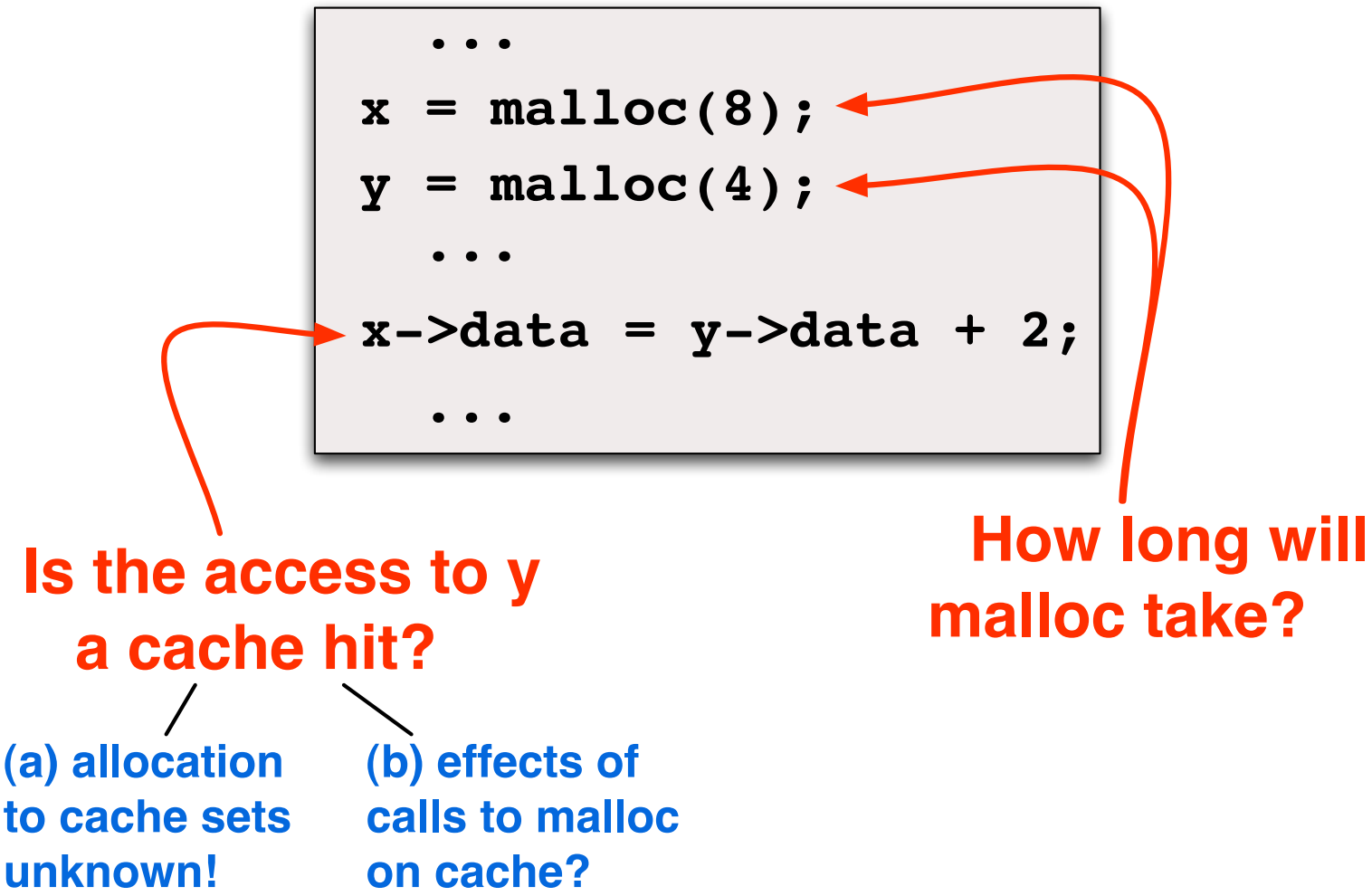
What we have ...

1. Precise WCET analysis
2. Dynamic Memory Allocation
   - often clearer program structure
   - easy memory reuse (e.g. in-situ transformations)

... but can we have both together?

Dynamic Memory Allocation & WCET Analysis

What are the challenges?

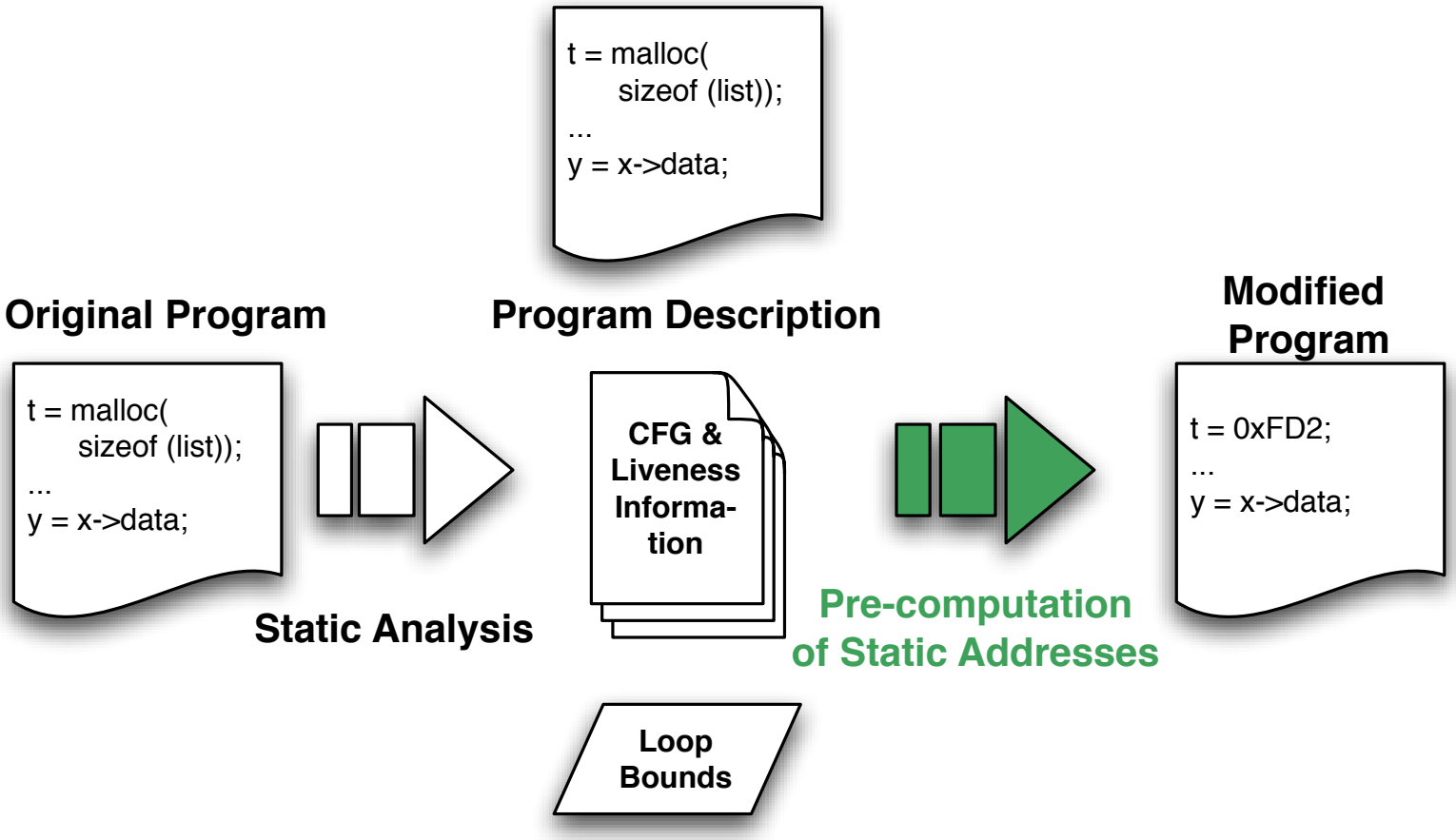


Approaches

What can be done?

- Predictable Allocator
  - J. Herter, J. Reineke, R. Wilhelm, 2008
- Predictable Hardware
  - M. Schöberl, 2009
- Replace Dynamic Allocation by Static Allocation

Big Picture


t = malloc(
sizeof (list));
...
y = x->data;
Original Program
Program Description
Modified Program
t = malloc(
sizeof (list));
...
y = x->data;
CFG & Liveness Informa-tion
t = 0xFD2;
...
y = x->data;
Static Analysis
Pre-computation of Static Addresses
Loop Bounds

What are *Good* Addresses?

What do we consider good addresses for heap allocated objects?

- Good addresses enable a subsequent WCET analysis to calculate minimal WCET bounds ...
- ... by minimal memory consumption.

How to find good addresses?

Idea for an algorithm:

1. Compute addresses s.t. memory consumption is minimal
   - Can generate an ILP to compute memory-optimal addresses from liveness information!

How to find good addresses?

Idea for an algorithm:

1. Compute addresses s.t. memory consumption is minimal
   - Can generate an ILP to compute memory-optimal addresses from liveness information!
2. Compute WCET for current addresses
   - Can generate an ILP to compute WCET from control-flow graph, loop bounds, and basic block information! (IPET, Li&Malik)

How to find good addresses?

Idea for an algorithm:

1. Compute addresses s.t. memory consumption is minimal
   - Can generate an ILP to compute memory-optimal addresses from liveness information!
2. Compute WCET for current addresses
   - Can generate an ILP to compute WCET from control-flow graph, loop bounds, and basic block information! (IPET, Li&Malik)
3. Select the program block with highest contribution to WCET

How to find good addresses?

Idea for an algorithm:

1. Compute addresses s.t. memory consumption is minimal
   - Can generate an ILP to compute memory-optimal addresses from liveness information!
2. Compute WCET for current addresses
   - Can generate an ILP to compute WCET from control-flow graph, loop bounds, and basic block information! (IPET, Li&Malik)
3. Select the program block with highest contribution to WCET
4. Modify addresses s.t. WCET contribution of selected block is minimized
   - Can generate an ILP to compute block-optimal addresses from liveness and basic block information!

How to find good addresses?

Idea for an algorithm:

1. Compute addresses s.t. memory consumption is minimal
   - Can generate an ILP to compute memory-optimal addresses from liveness information!
2. Compute WCET for current addresses
   - Can generate an ILP to compute WCET from control-flow graph, loop bounds, and basic block information! (IPET, Li&Malik)
3. Select the program block with highest contribution to WCET
4. Modify addresses s.t. WCET contribution of selected block is minimized
   - Can generate an ILP to compute block-optimal addresses from liveness and basic block information!
5. Continue at 2

Heuristics, actually ...

- ILPs to compute memory and block optimal addresses can be replaced by simulated annealing algorithms to cope with
  - more complex hardware (more cache sets)
  - more complex software (more heap allocated blocks)

Conclusions & Future Work

- Algorithm to compute static memory addresses for heap allocated objects.
- Preliminary experiments suggest feasible computational costs for addresses.

- Future Work:
  - Static (pre-) analysis to collect needed information.
  - *Fully* Automatic Transformation.