WUCC: Joint WCET and Update Conscious Compilation for Cyber Physical Systems

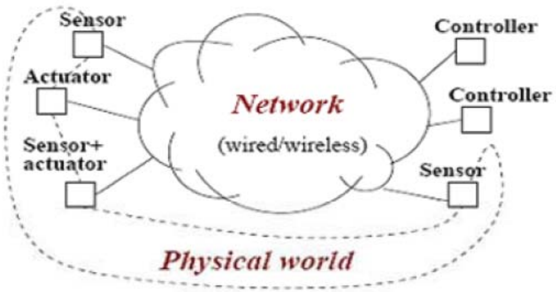
Yazhi Huang
Mengying Zhao
Chun Jason Xue

Department of Computer Science
City University of Hong Kong

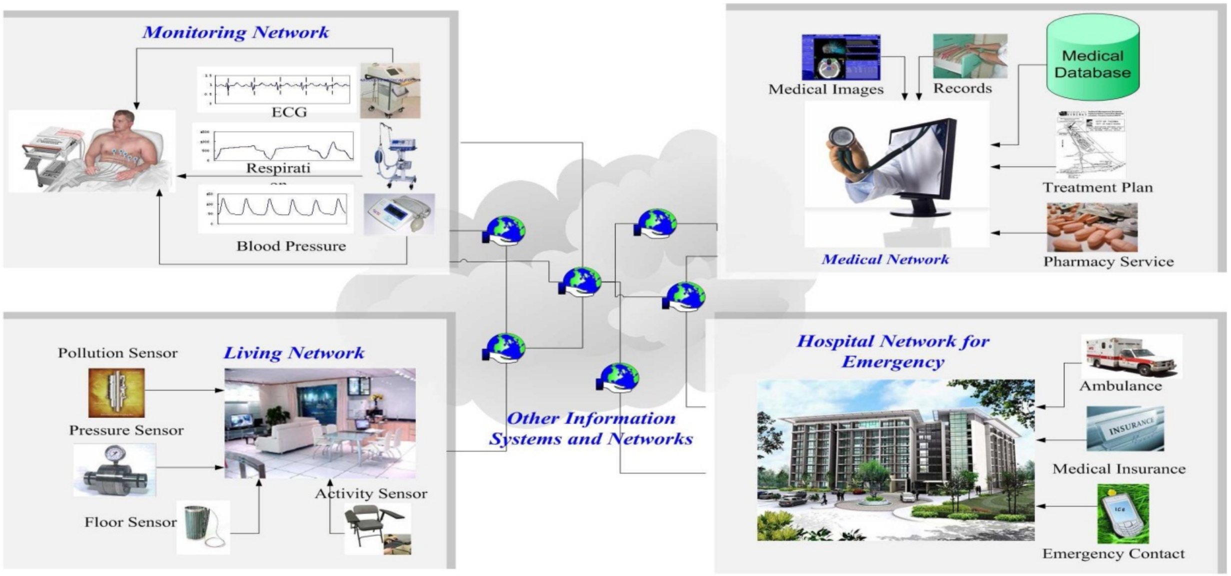
Outline

- ***Introduction***
- ***Motivational example***
- ***Problem analysis***
- ***The proposed algorithm***
- ***Experimental results***
- ***Conclusions***

Introduction



- *Cyber physical systems*
  - Usually real-time systems
  - A number of sensor nodes
  - Sensor nodes : powered by batteries, with preloaded code
  - Code update: wireless communication, energy consuming



Introduction

- *Challenges of cyber physical systems*

Reduce update

- **Code Update Problem**:
  - Update of preloaded code on remote sensor nodes powered by batteries is extremely energy consuming.

Reduce WCET

- **WCET problem**:
  - CPSs are often real-time embedded systems. Therefore, worst-case execution time (WCET) is an important real-time constraint

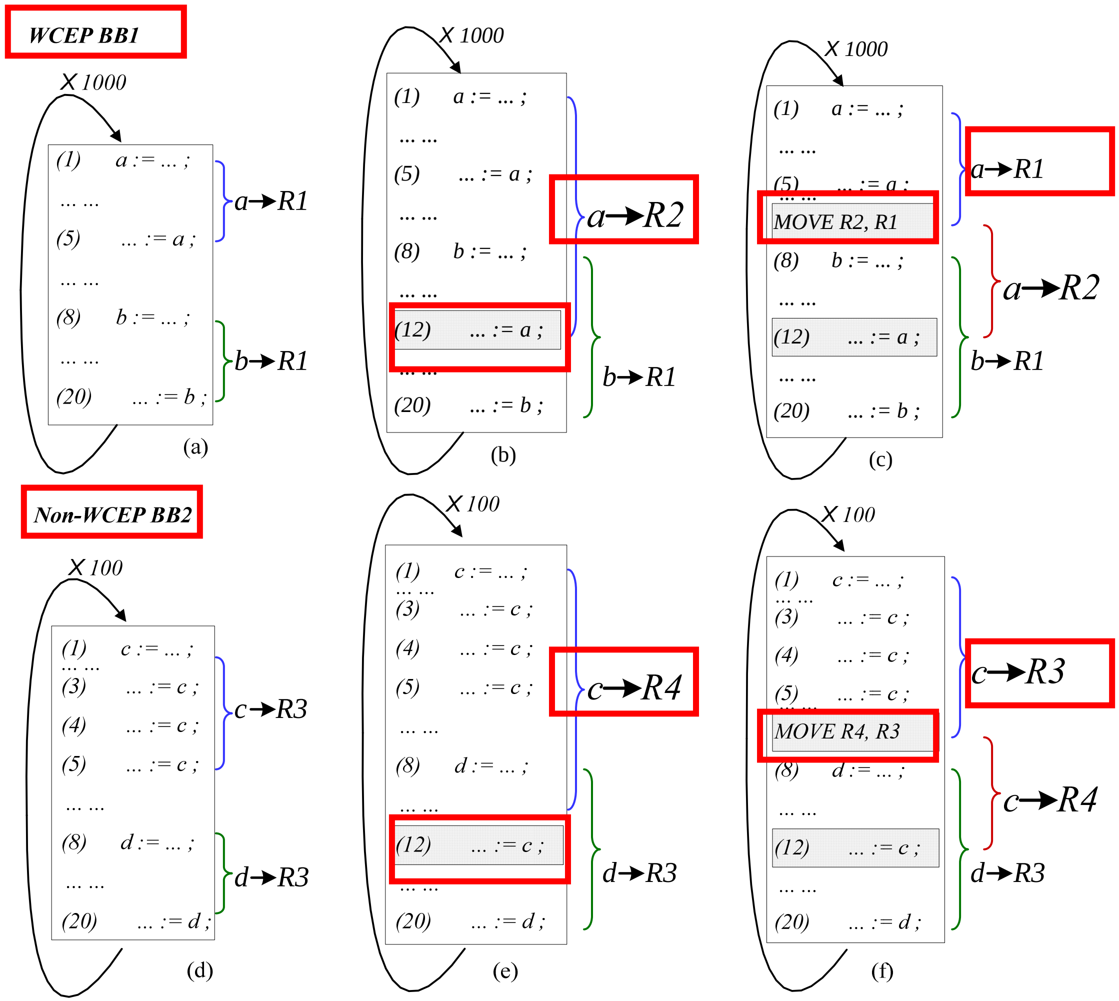
Previous works

- **Code Update Problem**: Li et al. proposed an update conscious compilation technique to improve the code similarity for energy consumption minimization in the wireless sensor network. ("UCC: update conscious compilation for energy efficiency in wireless sensor network" in *PLDI 07*)

- **WCET problem**: Falk presented a WCET-aware register allocator to avoid spill code generation along the critical path of a program for WCET reduction in real-time embedded systems. ("WCET-aware register allocation based on graph coloring" in *DAC 09*)

- **Limitations:**
  - UCC: Too many MOVE insertion → Increase in WCET
  - WCET-aware RA: Increase in code difference
  - Our goal is to reduce WCET and code difference simultaneously for real-time cyber physical systems

Motivational example



- BB1 is on WCEP while BB2 is not
- (a) (d) are original code
- (b) (e) are changed code
- (c) (f) are update conscious compilation solution
- With UCC technique, code similarity is improved by inserting necessary MOVE operations

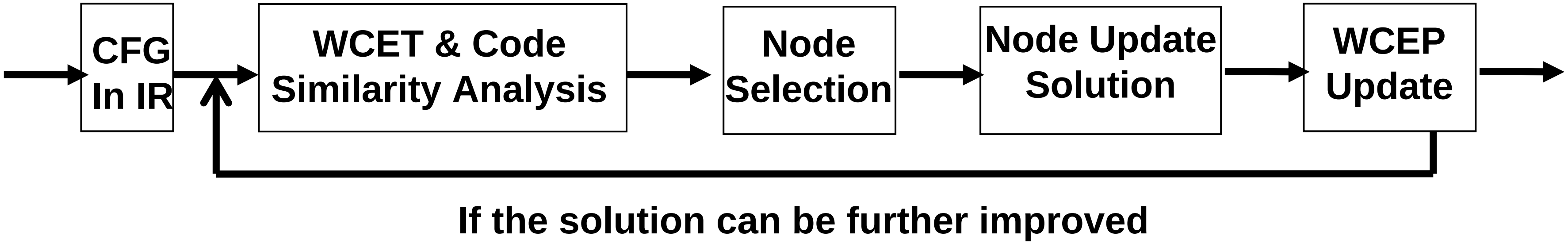
Motivational example

Approach	Negative WCET effect	Code diff	Energy saving
UCC [2]	100%	0%	100%
Proposed	0% - 10%	25%	75%
WCET [1]	0%	100%	0%

- By selecting appropriate basic blocks for UCC, most of code similarity may be achieved and at the same time have less negative effect on a program's WCET

- If implementing WCET-aware technique without considering code update, code similarity benefit is 0% (i.e. code difference is 100%)
- If only implementing UCC, relative code difference is 0%, but relative WCET increment is 100%
- The proposed technique only implements UCC for WCEP block BB1, but can improve most of code similarity, at 75% in this example

Overview



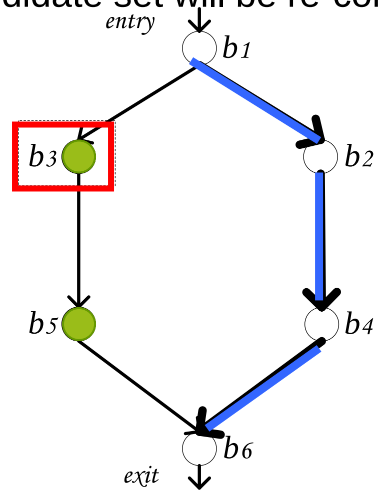
- CFG in IR form as input
- WCET analysis and code similarity analysis will be conducted simultaneously during the compilation process
- Each time select an appropriate CFG node
- Update-conscious compilation technique is implemented in the selected node
- New WCEP information is calculated and the new version code is used for next iteration of WCET and code similarity analysis
- This iteration continues until a balanced solution is obtained

Problem analysis

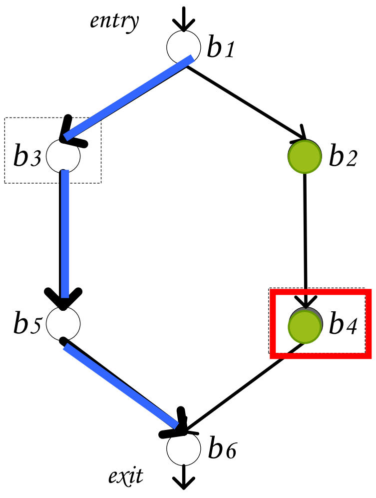
- *Strategy for CFG node selection*
  - Principle:
    - improve more node similarity
    - have zero or minimal negative effect on WCEP
  - Benefit:
    - processing this type of nodes first will leave more space for processing the rest of nodes
    - more nodes have potential to be selected and processed
  - Propose:
    - Therefore we propose to mark a less frequently executed node on non-WCEP with more number of executions and less variables to be updated for processing first

Problem analysis

- *Update candidate set during node selection*
  - WCEP change:
    - The candidate node set during node selection might change due to the potential change of WCEP after a block is processed
  - Candidate set re-construction:
    - If WCEP has changed, candidate set will be re-constructed
  - In the figure, execution path in bold is assumed to be the current WCEP
  - In figure (a), non-WCEP node b3 is select
  - In figure (b), candidate set is re-constructed, b4 becomes candidate and is selected



(a)



(b)

Problem analysis

- *A priority model for CFG node selection*

$$P_i = \frac{CS_i}{M_i \times Freq_i} \qquad (5)$$

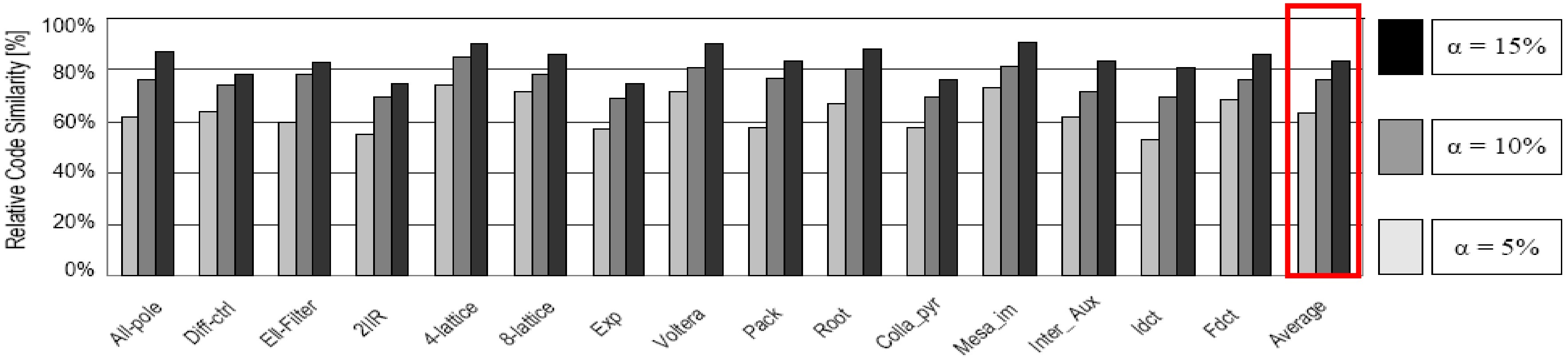
  - *Mi* : the number of *Move* operations that a node *i* requires.
  - *Freqi* : the execution frequency of a node *i*.
  - *CSi* : the code similarity benefit
  - Suggests : the more code similarity profit per unit potential increase in WCEP a node can bring, the higher priority it should be given
  - Benefit : less negative effect on WCET and more energy saving benefit

Algorithm

- *Set a default WCET increment counter*
- *Calculate WCEP*
- *Calculate priority*
- *Select a node based on priority*
- *Update conscious compilation technique is applied in the selected node*
- *Update counter*
- *Repeat above steps as long as WCET is less than a given threshold*

Experimental results

- *WCET Versus Code Similarity*



  - With a threshold of 10% increase in WCET, code similarity: 70% -- 85%. On average 76% of code similarity is achieved compared to UCC
  - 64% of code similarity can be achieved with 5% threshold while 84% benefit with 15% threshold
  - To summarize, with a small increase in WCET, WUCC can effectively achieve most of code similarity

Experimental results

- *Code difference among three approaches, WCET-aware technique as comparison base*

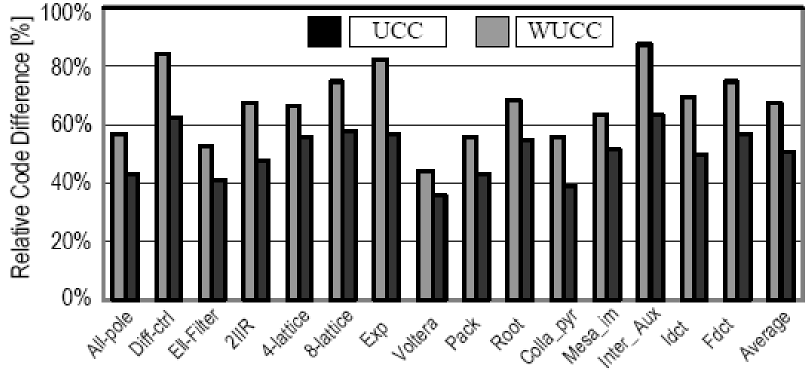


Fig. 7. The code difference of different benchmarks under WUCC, UCC, and WCET-aware technique. The results of WCET-aware compilation technique in [1] are used as the comparison base.

– With remarkable WCET benefit, code difference under WUCC is just a little larger than UCC

Conclusions

- We propose a compiler level optimization, joint WCET and update conscious compilation, for WCET and code difference minimization in cyber physical systems
- A novel CFG node selection heuristic is proposed, where a priority based model is built by considering a node's code similarity benefit, MOVE operation requirement, and the execution frequency
- We formulate the target problem and implement a greedy algorithm to achieve a balanced result

香港城市大學
City University
of Hong Kong

Thank you!

Thank you!