



Tighter WCET Estimates by Procedure Cloning

Paul Lokuciejewski, Heiko Falk, Martin Schwarzer, Peter Marwedel

Computer Science XII

University of Dortmund

D-44221 Dortmund

{Paul.Lokuciejewski, Heiko.Falk, Martin.Schwarzer, Peter.Marwedel}@udo.edu





Outline

1. Introduction
2. Procedure Cloning
3. Environment & Results
4. Future Work





1. Reducing Analysis Complexity

- loops are of main interest in ES software
- for precise static WCET analysis
 - ➔ represent each loop iteration by single context
- example MPEG: 1 context: 126 s ↔ 4 contexts: 4,5 h
- compromise: restrict distinguished contexts
 - summarize loop iterations into single context
 - ➔ high WCET overestimation

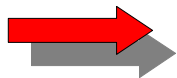




1. Insufficient Loop Annotations

- loops annotated by [min,max]
- problem: **loop executions controlled by func. parameters**
- no explicit specification of loop iterations possible
 - exact loop bounds cannot be derived
- overestimation for last summarized context
 - static WCET analysis must **assume worst case** (safeness)
 - always assume upper loop bounds & longest path
- solution for tight WCET estimates: **Procedure Cloning**

```
int f(int n) { ...  
  for (i=0; i <= n; ++i) {...} }  
}
```



loop analysis highly imprecise





2. Procedure Cloning

- also known as **Function Specialization**
- compiler optimization aiming at reducing ACET
- generates specialized function copies
 - provides good basis for inter-procedural DFAs
 - opportunity for further standard optimizations
 - reduces calling overhead
 - simplifies control flow

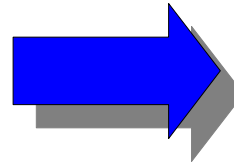




2. Example for Procedure Cloning

```
int f(float *x, int n, int p) {  
  for (i=0; i<=n; ++i) {  
    x[i] = p * x[i];  
    if(i==10) {...}  
  }  
  return x[n];  
}
```

```
int main(void) {  
  // multiple calls of f(x,5,2);  
  return f(a,5,2);  
}
```



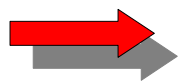
```
int f1(float *x) {  
  for (i=0; i<=5; ++i) {  
    x[i] = 2 * x[i];  
    if( i==10) {...}  
  }  
  return x[5];  
}
```

```
int main(void) {  
  // multiple calls of f1(x);  
  return f1(a);  
}
```



2. Exploit for WCET estimation

- Procedure Cloning makes code more predictable
 - function clones dedicated to individual loop executions
 - makes different function call contexts explicit
 - eliminates infeasible paths
- suitable for typical ES software
 - loop iterations controlled by function parameters
 - functions called multiple times with varying constants
- restriction of contexts (less) harmful



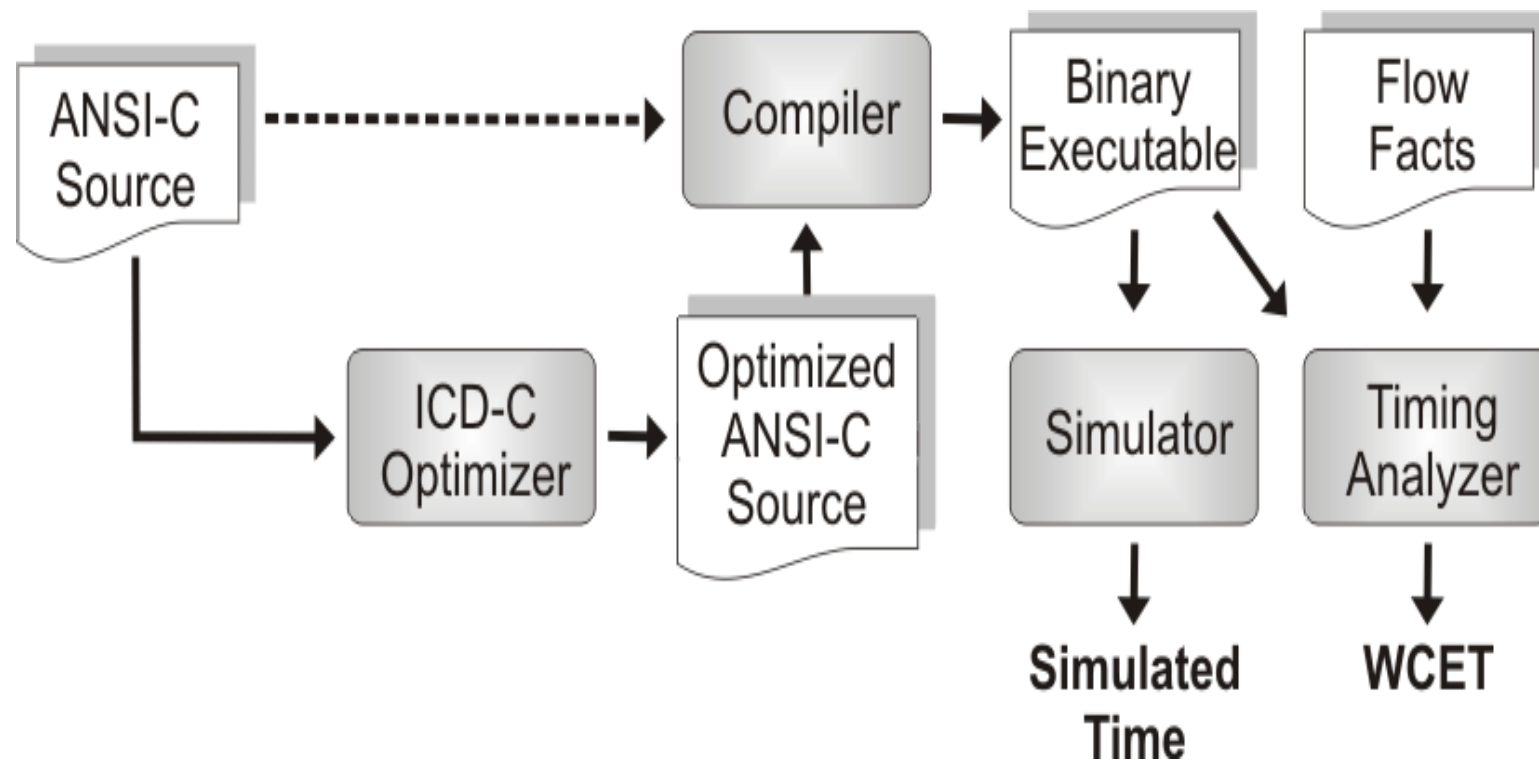
fast way to get tighter WCET estimates





3. Experimental Environment

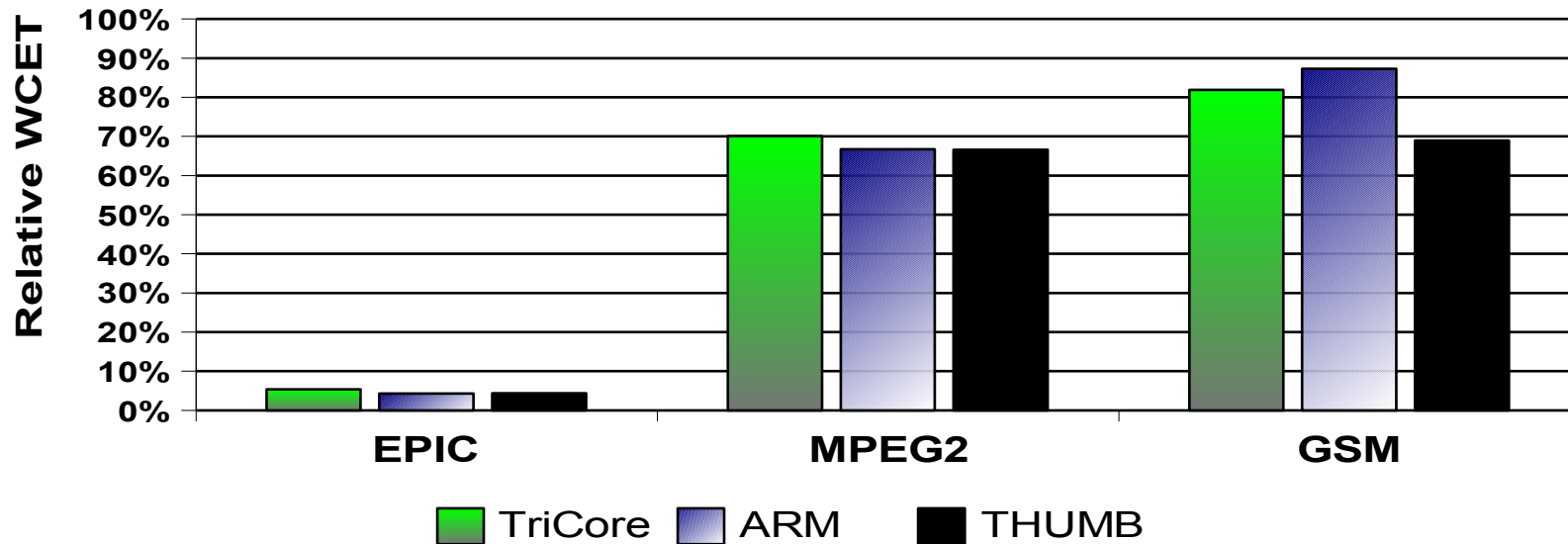
- benchmarks from MiBench suite
- analysis for TC1796 and ARM7TDMI (ARM+THUMB)
- AbsInt's aiT used as WCET analyzer





3. Results

- relative WCET for opt. code wrt. original code (100%)



- improvements for both processors of **up to 95 %**
 - results from nested loops controlled by constants
- code size increase: **up to 300%**
- increase of WCET analyzer runtime: **on average 14%**





4. Future Work

- incorporation into WCET-aware C compiler **WCC**
- better exploitation of Procedure Cloning
 - optimize functions on WC path
- cope with increasing code size
 - perform a trade-off between WCET and code size
- automatic generation of flow facts
- virtual Procedure Cloning at analysis level

