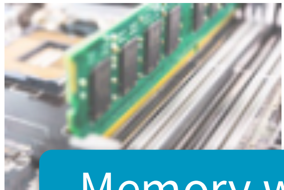
Parallel Real-Time Tasks, as viewed by WCET Analysis and Task Scheduling Approaches

Christine Rochange

16th Workshop on Worst-Case Execution Time Analysis
5th July, 2016

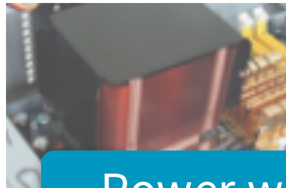
Motivation

The performance of single-core processors does not grow anymore



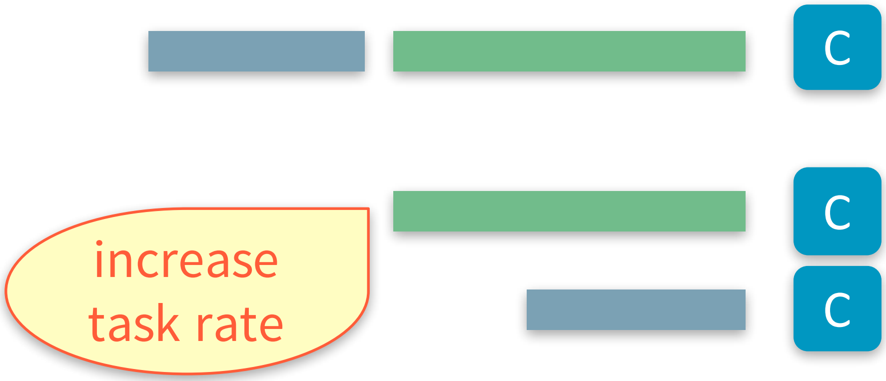
Memory wall

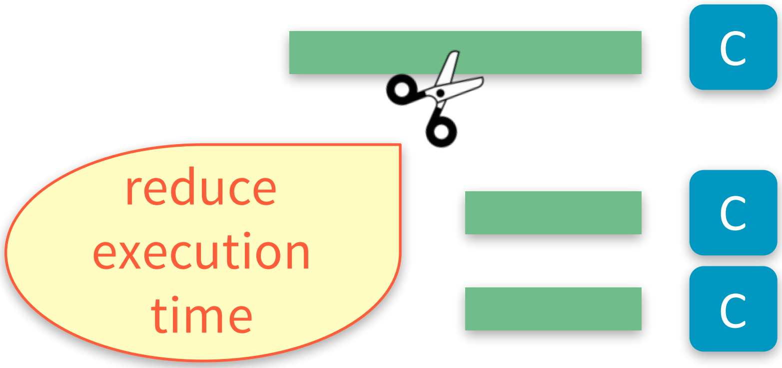
ILP wall



Power wall

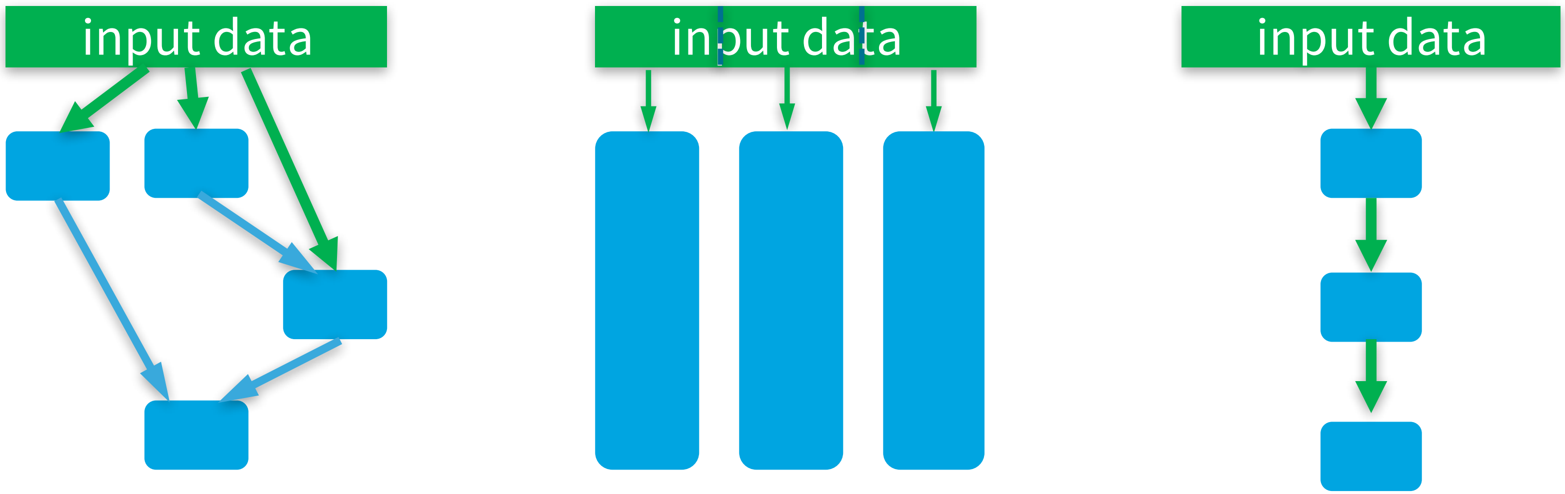
Multi-core processors have become the norm





Different flavours of parallel programming

Task vs. data parallelism



Interactions between threads

- exchange of data (message passing, mutual exclusion, etc.)
- progress synchronisation

Disclaimer

Hardware-level interferences are out of the scope

- they may have a strong impact on WCETs
  - bandwidth-sharing (bus, memory controller, etc.)
  - space-sharing (e.g. shared L2 cache)

- research on how this impact can be accounted for is ongoing
  - still some open questions

  BUT
  - let's assume the problem solved (analysable/predictable platform)

- an orthogonal issue wrt. analysing software-level interferences

Literature

Timing analysis of parallel real-time tasks

Do they all speak of the same *parallel* tasks?

How do they fit together?

WCET analysis of parallel tasks

Background on WCET analysis for *sequential* tasks

TASK UNDER ANALYSIS

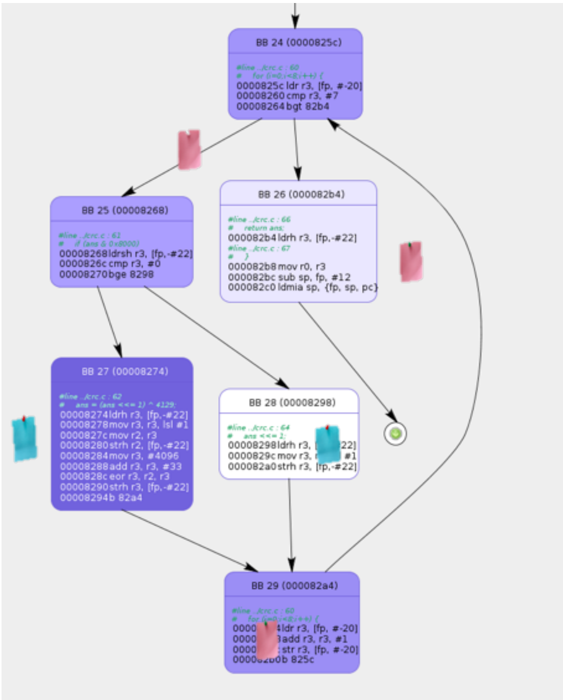
source code

binary code

STATIC ANALYSES

FLOW ANALYSES

LOW-LEVEL ANALYSES



WCET COMPUTATION

Integer Linear Programming

WCET upper-bound

WCET analysis of parallel tasks

Example 1

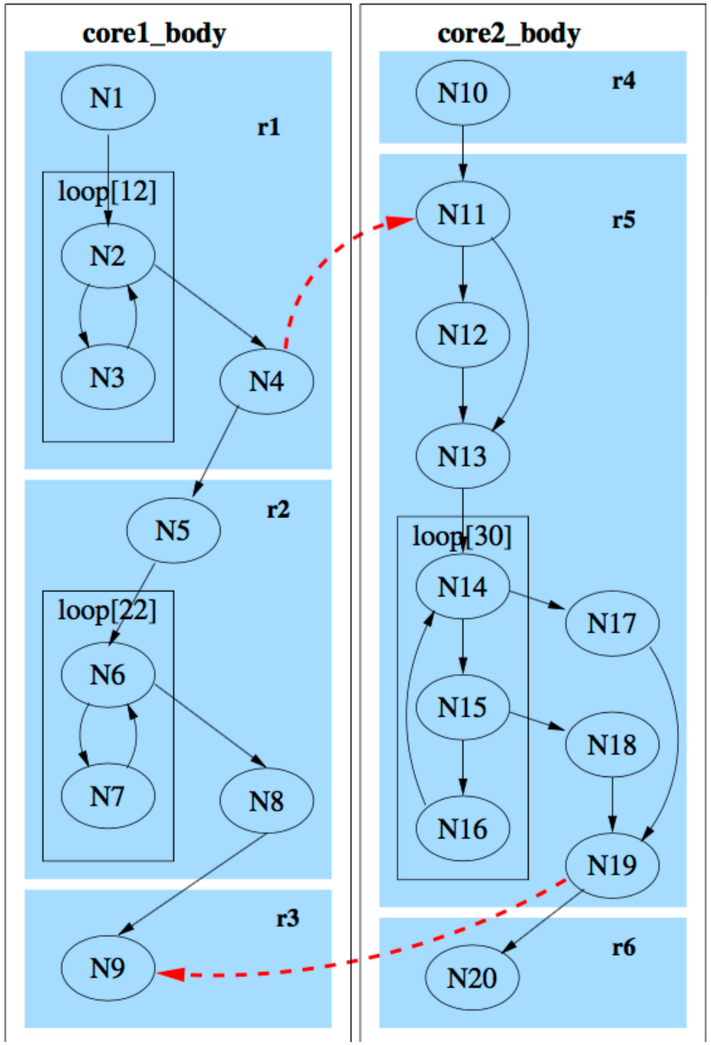
```
void core1() {
  int tqmf[24]; long xa, xb, el;
  int xin1, xin2, decis_levl;
  for(;;) { //Infinite loop
    // Computation phase 1
    xa = 0; xb = 0;
    for (i=0;i<12;i++) { // 12 iterations
      xa += (long) tqmf[2*i]*h[2*i];
      xb += (long) tqmf[2*i+1]*h[2*i+1];
    }
    // Send the results to core 2
    send(channel1,(int)((xa+xb)>>15)); ------
    // Read inputs
    xin1=read_input(); xin2=read_input();
    // Computation phase 2
    for(i=23;i>=2;i--) { // 22 iterations
      tqmf[i]=tqmf[i-2];
    }
    tqmf[1] = xin1; tqmf[0] = xin2;
    // Receive data from core2 and output it
    decis_levl = receive(channel2) ; <------
    write_output(decis_levl) ;
  }
}
```

```
const int decis_levl [30];
int core2() {
  int q,el;

  for(;;) {//Infinite loop

    // Receive data from core1
--> el = receive(channel1);
    // Computation phase 1
    el = (el>=0)?el:(-el);
    for (q = 0; q < 30; q++) {
      // 30 iterations
      if (el <= decis_levl[q])
        break;
    }
    // Send result to core1
--- send(channel2,decis_levl) ;

  }
}
```

D. Potop-Butucaru, I. Puaut - **Integrated WCET Estimation of Multicore Applications**
Workshop on WCET Analysis, 2013

WCET analysis of parallel tasks

Example 1



Joined CFGs

- additional edges to express precedence

```
void core1() {
  int tqmf[24]; long xa, xb, el;
  int xin1, xin2, decis_levl;
  for(;;) { //Infinite loop
    // Computation phase 1
    xa = 0; xb = 0;
    for (i=0;i<12;i++) { // 12 iterations
      xa += (long) tqmf[2*i]*h[2*i];
      xb += (long) tqmf[2*i+1]*h[2*i+1];
    }
    // Send the results to core 2
    send(channel1,(int)((xa+xb)>>15)); ———
    // Read inputs
    xin1=read_input(); xin2=read_input();
    // Computation phase 2
    for(i=23;i>=2;i--) { // 22 iterations
      tqmf[i]=tqmf[i-2];
    }
    tqmf[1] = xin1; tqmf[0] = xin2;
    // Receive data from core2 and output it
    decis_levl = receive(channel2) ; <———
    write_output(decis_levl) ;
  }
}
```

```
const int decis_levl [30];
int core2() {
  int q,el;

  for(;;) {//Infinite loop

    // Receive data from core1
———> el = receive(channel1);
    // Computation phase 1
    el = (el>=0)?el:(-el);
    for (q = 0; q < 30; q++) {
      // 30 iterations
      if (el <= decis_levl[q])
        break;
    }
    // Send result to core1
——— send(channel2,decis_levl) ;
  }
}
```


D. Potop-Butucaru, I. Puaut - **Integrated WCET Estimation of Multicore Applications**
Workshop on WCET Analysis, 2013

WCET analysis of parallel tasks

Example 2

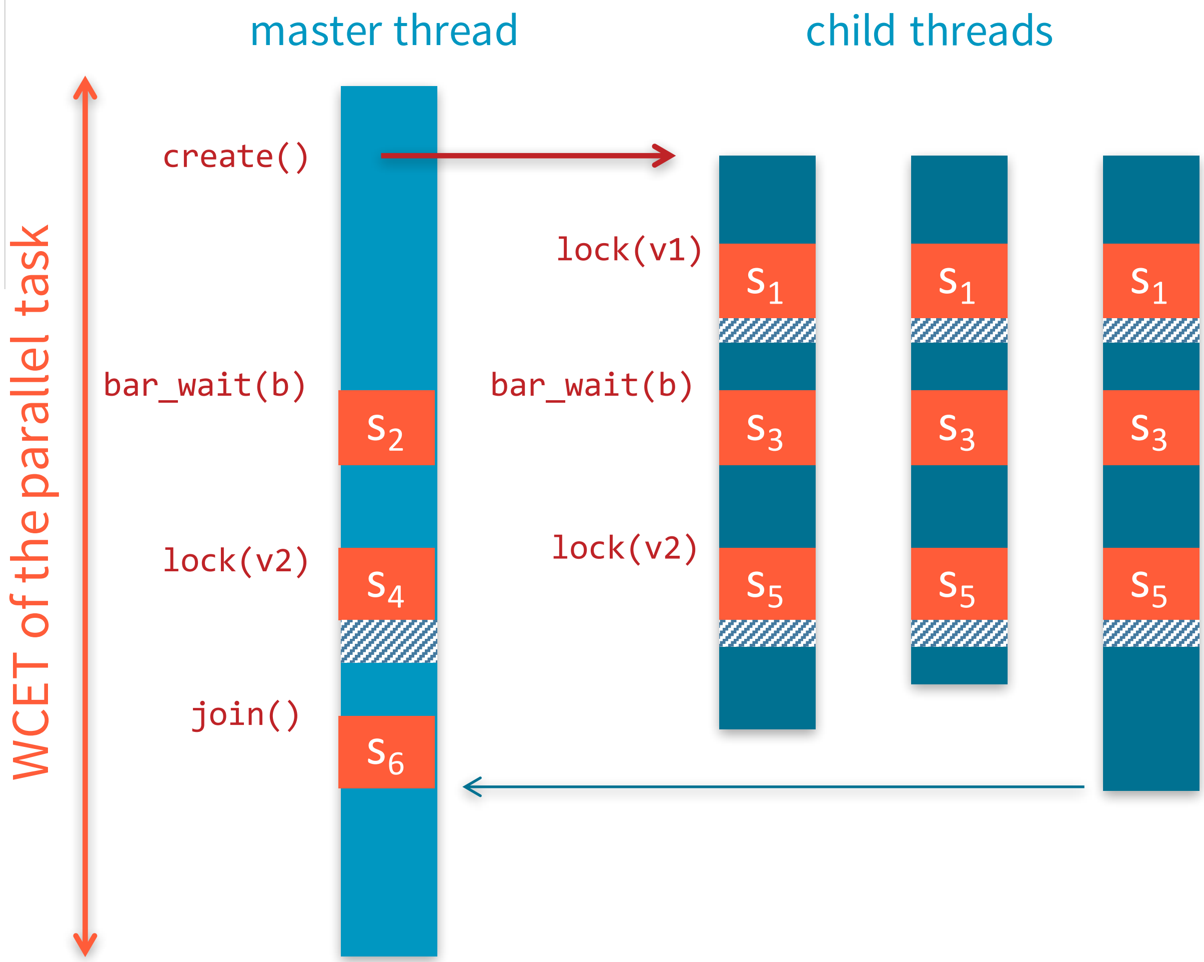
```
int main() {
  for (int i=0; i<2; i++)
    CREATE_THREAD(&work);
  ...
  BARRIER(&bar,3);   // ID=bar
  ...
  for (int i=0; i<2; i++)
    JOIN(i+1);   // ID=join
}
```

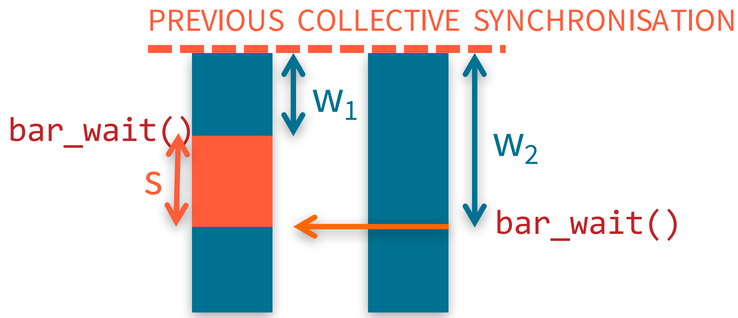
```
void work() {
  ...
  BARRIER(&bar,3);   // ID=bar
  ...
  MUTEX_LOCK(&lock);   // ID=cs
  ... // critical section
  MUTEX_UNLOCK(&lock);   // ID=cs
  ...
}
```

H. Ozaktas, C. Rochange, P. Sainrat - **Automatic WCET Analysis of Real-Time Parallel Applications** - Workshop on WCET Analysis, 2013

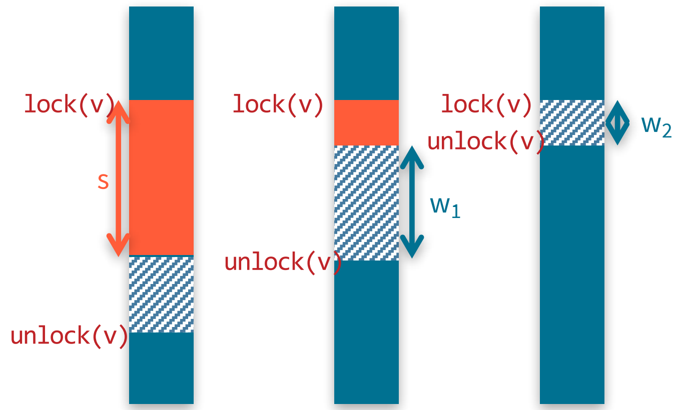
WCET analysis of parallel tasks

Example 2





$$s \leq \max(0, w_2 - w_1)$$

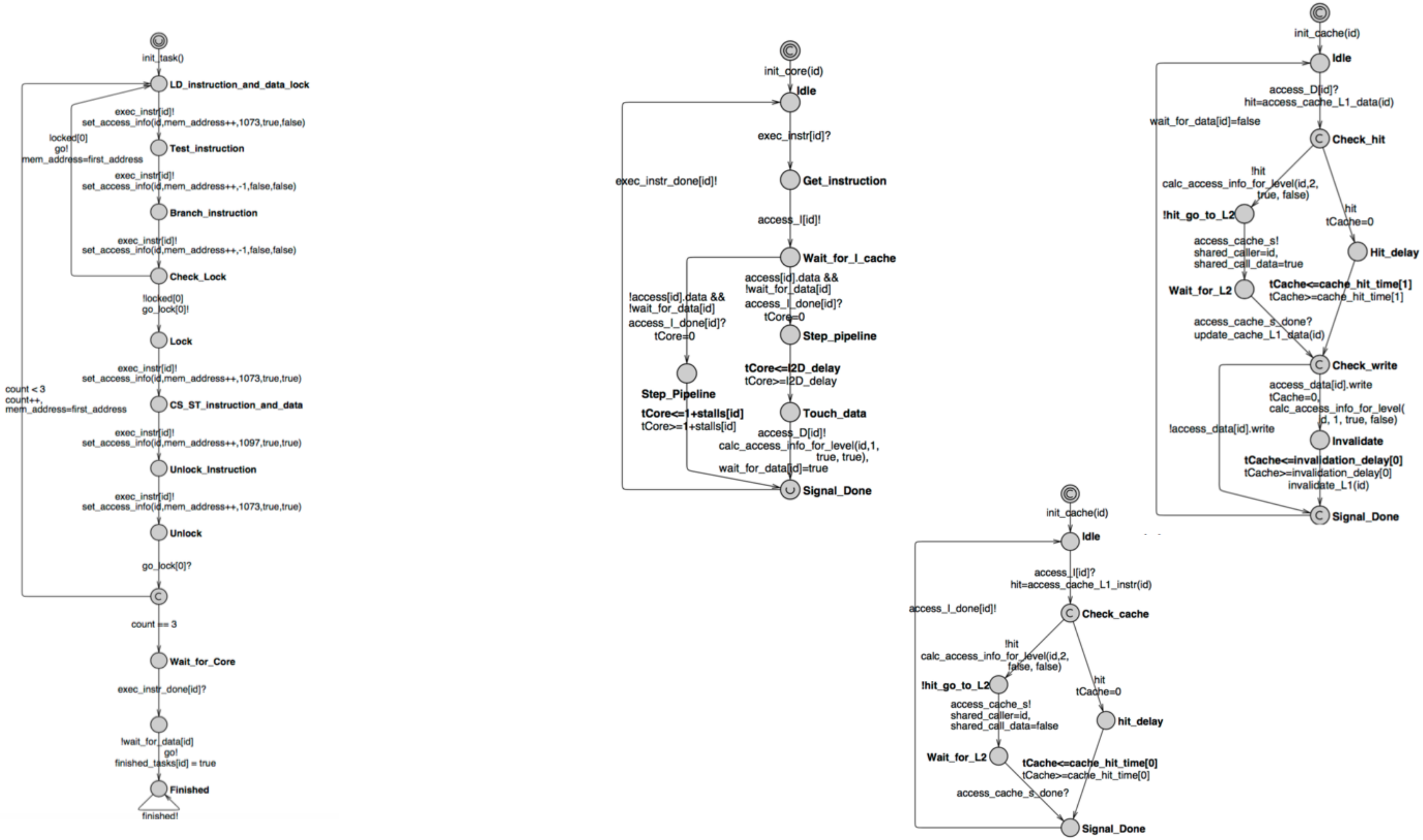


$$s \leq w_1 + w_2$$

H. Ozaktas, C. Rochange, P. Sainrat - **Automatic WCET Analysis of Real-Time Parallel Applications** - Workshop on WCET Analysis, 2013

WCET Analysis of Parallel Tasks

Example 3

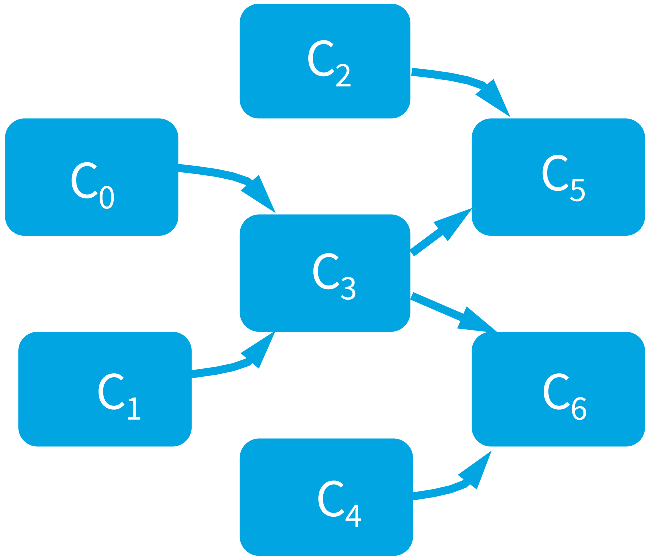


A. Gustavsson, A. Ermedahl, B. Lisper, P. Pettersson - **Towards WCET Analysis of Multicore Architectures Using UPPAAL** - Workshop on WCET Analysis, 2010

Scheduling parallel tasks

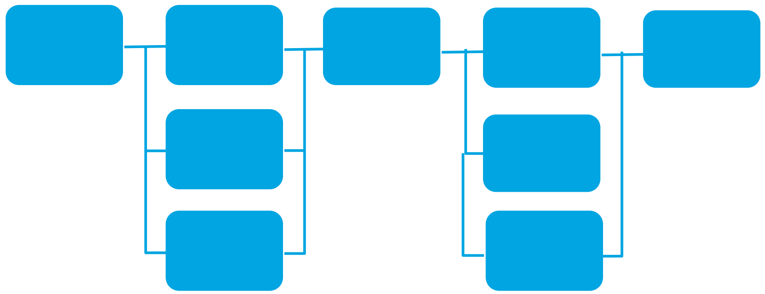
Parallel task models

P, D, ~~C~~

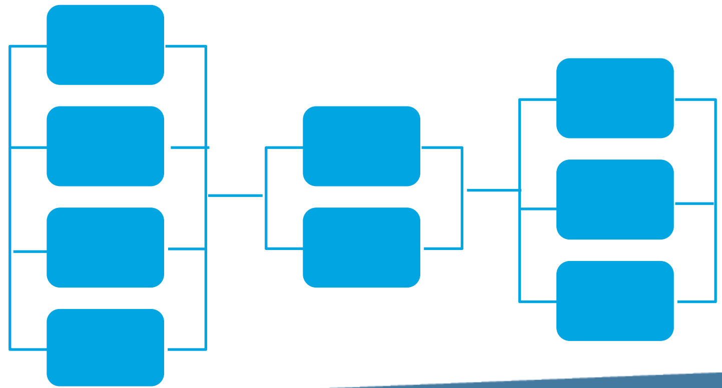


DAG

fork-join



multi-frame segment

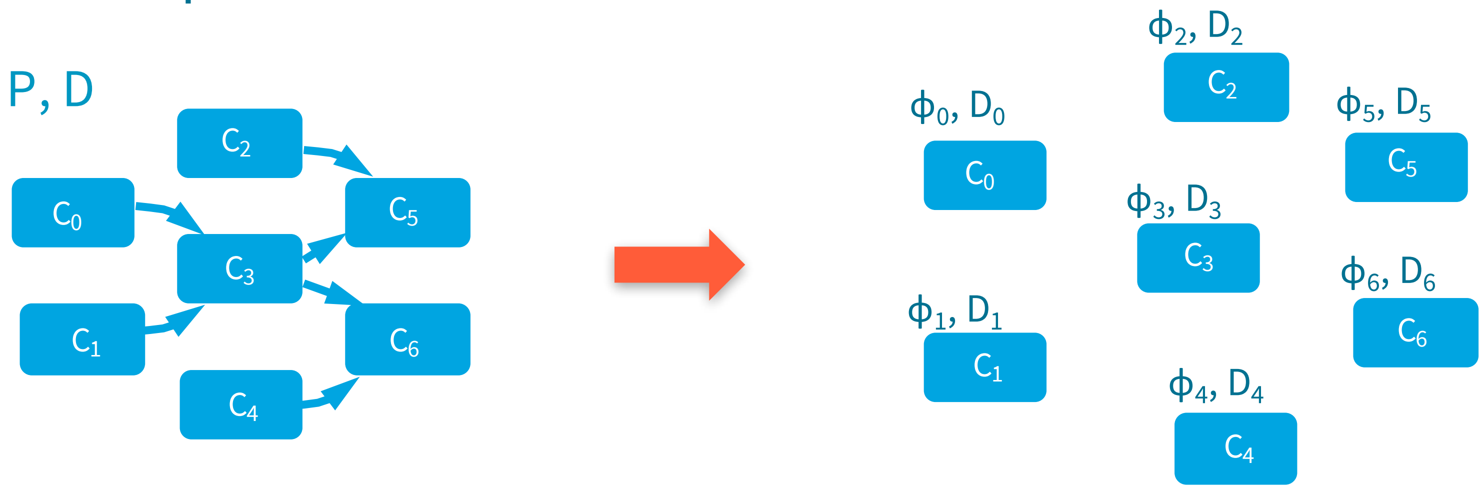


Scheduling parallel tasks

Two kinds of approaches
- DAG transformation
- Direct scheduling

DAG transformation

- transforms a parallel task (DAG of subtasks) into a set of independent sequential tasks



- schedules independent tasks using techniques for scheduling sequential tasks on multicores

Saifullah et al., **Multicore Real-Time Scheduling for Generalized Parallel Task Models**, J. RTS, 2013
Qamieh et al., **A Stretching Algorithm for Parallel Real-Time DAG Tasks**, RTNS 2014

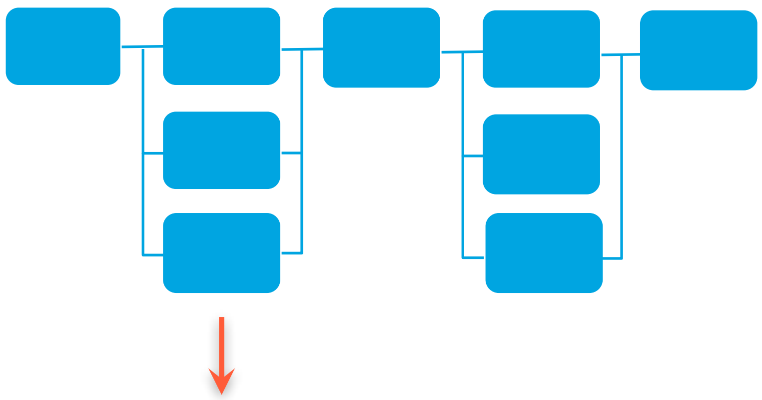
Scheduling parallel tasks

Two kinds of approaches

- DAG transformation
- Direct scheduling

Direct scheduling

- Gang scheduling



one thread per core
threads run simultaneously

Kato et al., **Gang EDF Scheduling for Parallel Task Systems**, RTSS, 2009

Scheduling parallel tasks

Two kinds of approaches

- DAG transformation
- Direct scheduling

Direct scheduling

- Gang scheduling
- Federated scheduling
  - allocates a dedicated cluster of cores
  - schedules subtasks with a greedy strategy

$$n_i = \left\lceil \frac{C_i - L_i}{D_i - L_i} \right\rceil$$

Li et al., **Analysis of Federated and Global Scheduling for Parallel Real-Time Tasks**, ECRTS, 2014

Scheduling parallel tasks

Two kinds of approaches

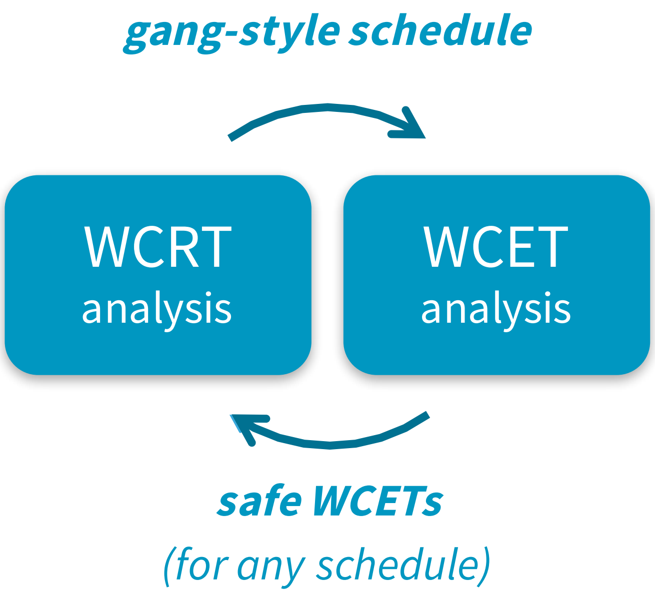
- DAG transformation
- Direct scheduling

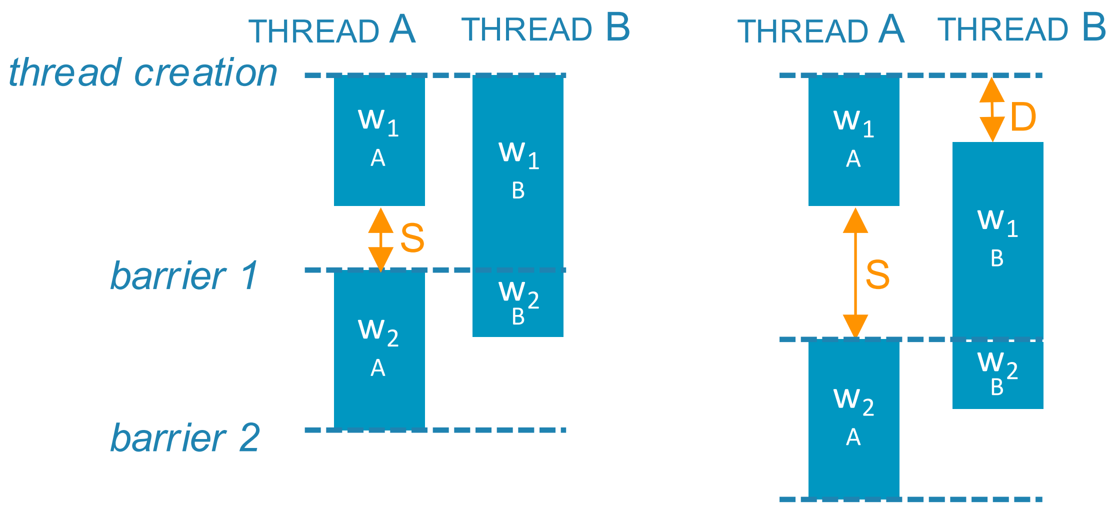
Direct scheduling

- Gang scheduling
- Federated scheduling
- Global EDF scheduling
  - schedules ready nodes with the earliest deadline

Baruah et al., **A Generalized Parallel Task Model for Recurrent Real-Time Processes**, RTSS, 2012

Key assumptions





Summary

Different focus

- independent subtasks with precedence constraints *sched./WCRT*
  vs.
  communicating/synchronising subtasks *WCET*

Idealized assumptions

- schedule-independent subtask WCETs *sched./WCRT*
  vs.
  gang-like schedule *WCET*

Need for tighter cooperation

Preliminary ideas

- Schedule-dependent WCET analysis
  - release offsets for parallel threads
  - safer/tighter estimations of worst-case stall times
- Enriched task model
  - scheduling directives/constraints
  - schedule-dependent WCETs

- Scheduling/WCET-aware parallel programming


TIME FOR QUESTIONS