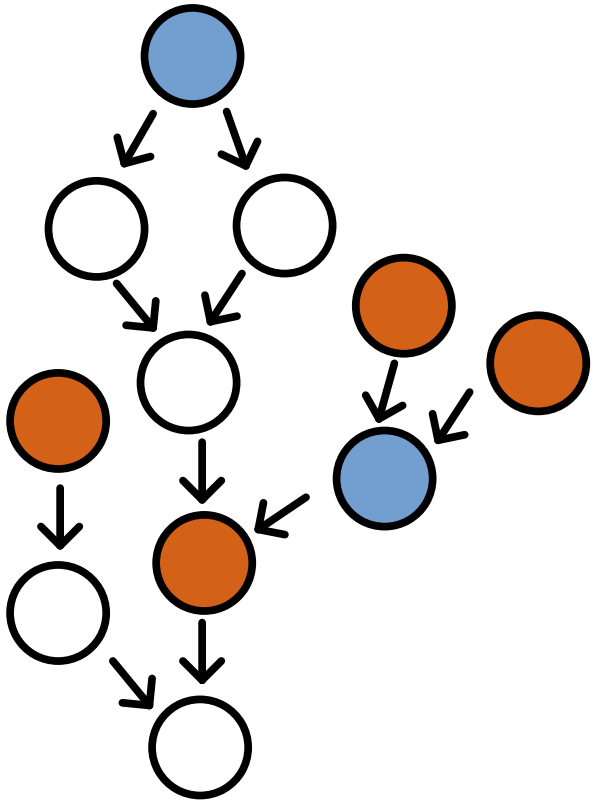
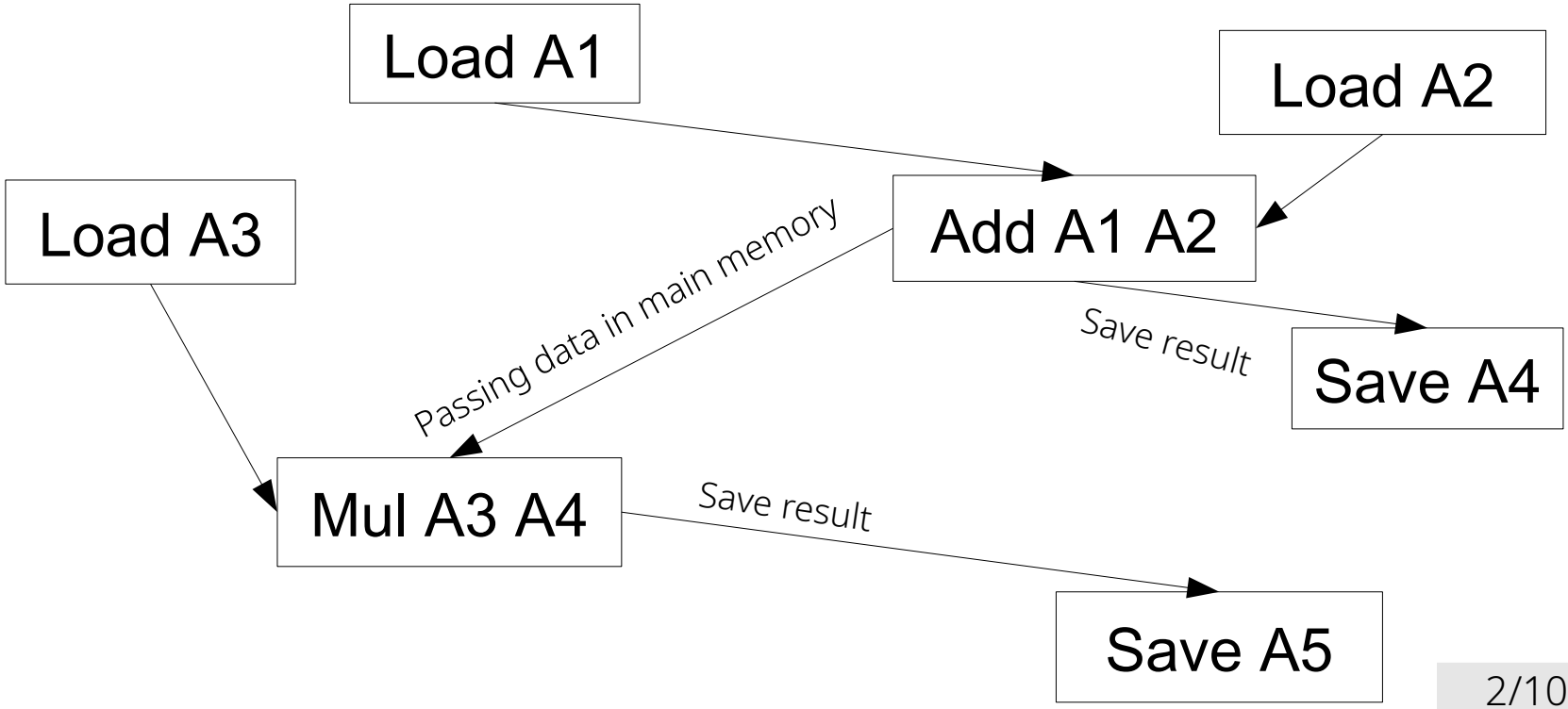
OpenMP Tasking

Piotr Luszczek

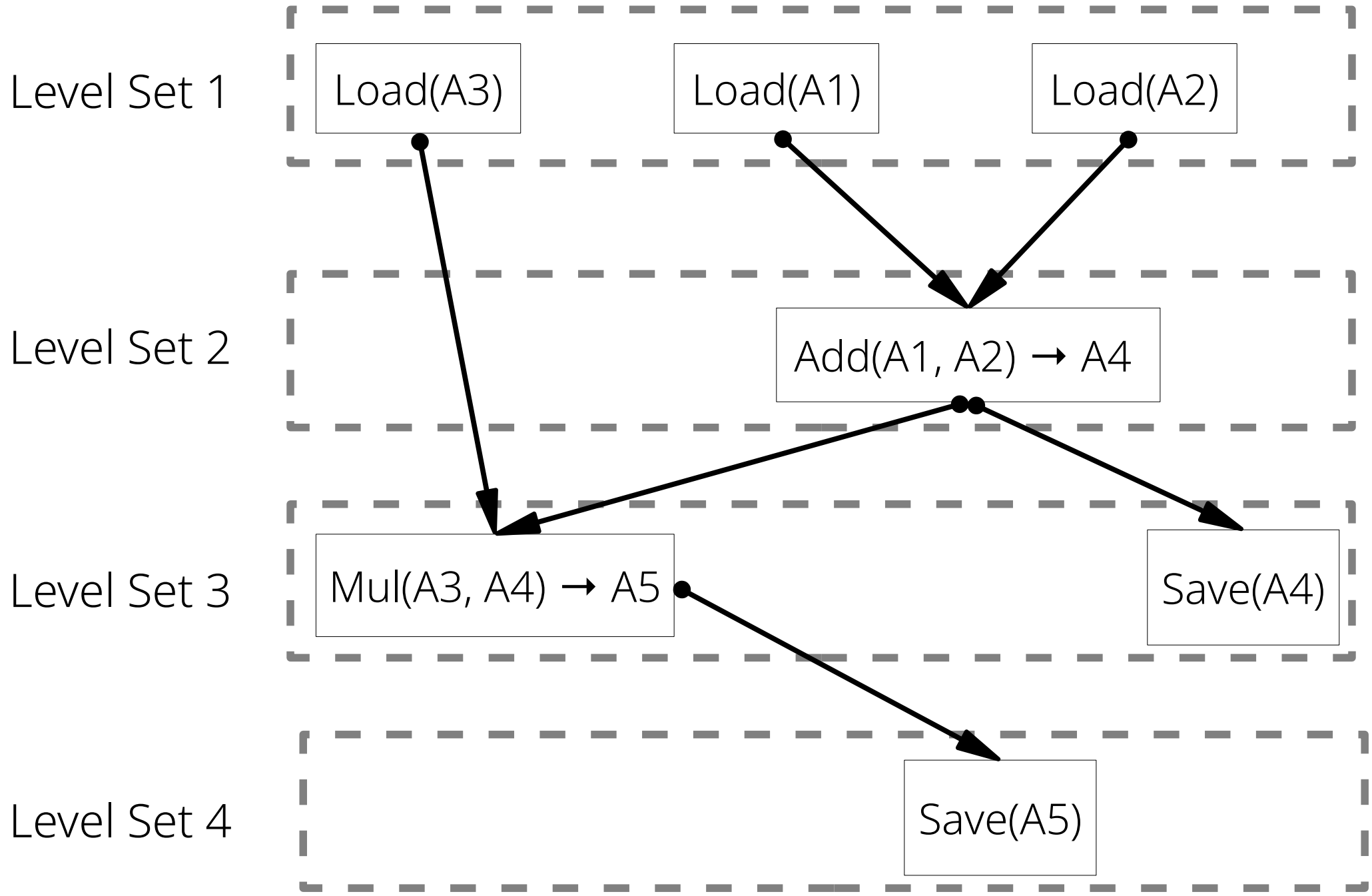
Representing Code as DAG of Tasks



```
double A1[N], A2[N], A3[N], A4[N], A5[N];
Load(A1, N);
Load(A2, N);
Load(A3, N);
Add(N, A1, A2, A4);
Mul(N, A3, A4, A5);
Save(A4);
Save(A5);
```



DAG of Tasks and Its Level Sets



OpenMP Tasking Overview

- OpenMP with version 4 started to support tasking with data dependences
- Tasks were part of OpenMP since version 3
  - These were called sibling tasks and available in Cilk
- Version 4 introduced data dependence clause
  - New clause "depend" allows to constraint execution of tasks based on how they pass data between each other
  - The data dependence also allows the tasks to synchronize selectively between each other based on the data they exchange
- OpenMP runtime scheduler decides which cores to run the tasks on
  - The scheduling decisions are based on:
    - Amount of work each core has (work stealing for balancing)
    - Availability of data in caches
      - It is beneficial to use the same core for tasks that share data that is already in cache (affinity)

Dynamic Scheduling, OpenMP, and GNU GCC

Date	OpenMP	GCC	Pragma	Clause
2008 May	3.0		#pragma omp task	
2009 April		4.4		
2013 July	4.0		#pragma omp task	depend
2014 April		4.9		
2015 November	4.5		#pragma omp task	priority
2016 April		6.1		
2018 November	5.0		#pragma omp task	affinity (location) detach(handle)
2019 May		9.1		
2019 November	5.1 preview			

Task Directive Summary

- #pragma omp task
  - if (expression)
  - final
  - untied
  - default
  - mergeable
  - private (list)
  - firstprivate (list)
  - shared (list)
  - depend (dependence type : list)
  - priority (expression)

```
#pragma omp parallel
{ // only master thread creates tasks
  #pragma omp master
  {
    partition(array); // sequential

    #pragma omp task
    qsort(array, n/2);

    #pragma omp task
    qsort(array+n/2, n/2);

    #pragma omp taskwait // wait for tasks

    merge(array, array+n/2, n); // sequential
  }
}
```

- task directive is the most complicated among the common ones
- The most important clauses will be discussed next

Depend Clause Summary

- #pragma omp task depend(*dependence type : list*)
- Dependence type can be either
  - in
    - Use it for data that is consumed by the tasks
  - out
    - Use it for data that is produced by the tasks
  - inout
    - Use it for data that is both consumed and produced by the tasks
  - Fortran specification for subroutine function parameters
    - Since Fortran 90: in, out, inout, scratch
- List is a comma-separated enumeration of variables participating in data dependence graph
  - Arrays are specified with ranges:
    - For example: depend(in:A[0:N])

Implementation of the Sample Task DAG

```
#pragma omp parallel shared(a1, a2, a3, a4, a5, n)
{
 #pragma omp master
 {
   #pragma omp task depend(out:a1[0:n])
   Load(n, a1);
   #pragma omp task depend(out:a2[0:n])
   Load(n, a2);
   #pragma omp task depend(out:a3[0:n])
   Load(n, a3);
   #pragma omp task depend(in:a1[0:n],a2[0:n]) depend(out:a4[0:n])
   Add(n, a1, a2, a4);
   #pragma omp task depend(in:a3[0:n],a4[0:n]) depend(out:a5[0:n])
   Mul(n, a3, a4, a5);
   #pragma omp task depend(in:a4[0:n])
   Save(n, a4);
   #pragma omp task depend(in:a5[0:n])
   Save(n, a5);
 }
}
```

- The order in which tasks are inserted is important
  - To be safe, exhaust the tasks from the first level set before inserting tasks from the second level

Advanced Use of Tasks

- It is possible to make complicated task graphs
  - taskwait
    - directive creates a join point of descendent tasks of the current task
  - taskgroup
    - allows creating of and waiting for descendant tasks
  - taskyield
    - directive allows to relinquish CPU for other tasks
  - taskloop
    - directive allows generation of tasks with loop constructs

OpenMP 4+ is not the Only One for Tasking

- Task-based data-parallel computing has become an important method for parallel computing
  - It has grown tremendously since the multicore era began
- There are many projects that use this paradigm
  - Shared memory
    - Hstreams, Open Community Runtime, ompSS, QUARK, Thread Building Blocks
  - Distributed memory
    - Legion, PaRSEC, RAJA, StarPU, Thor
  - There are more projects that provide this functionality