Integration of Transactional Memory to a data-flow, streaming extension of OpenMP

Ismail KURU[1]

Albert COHEN[2]

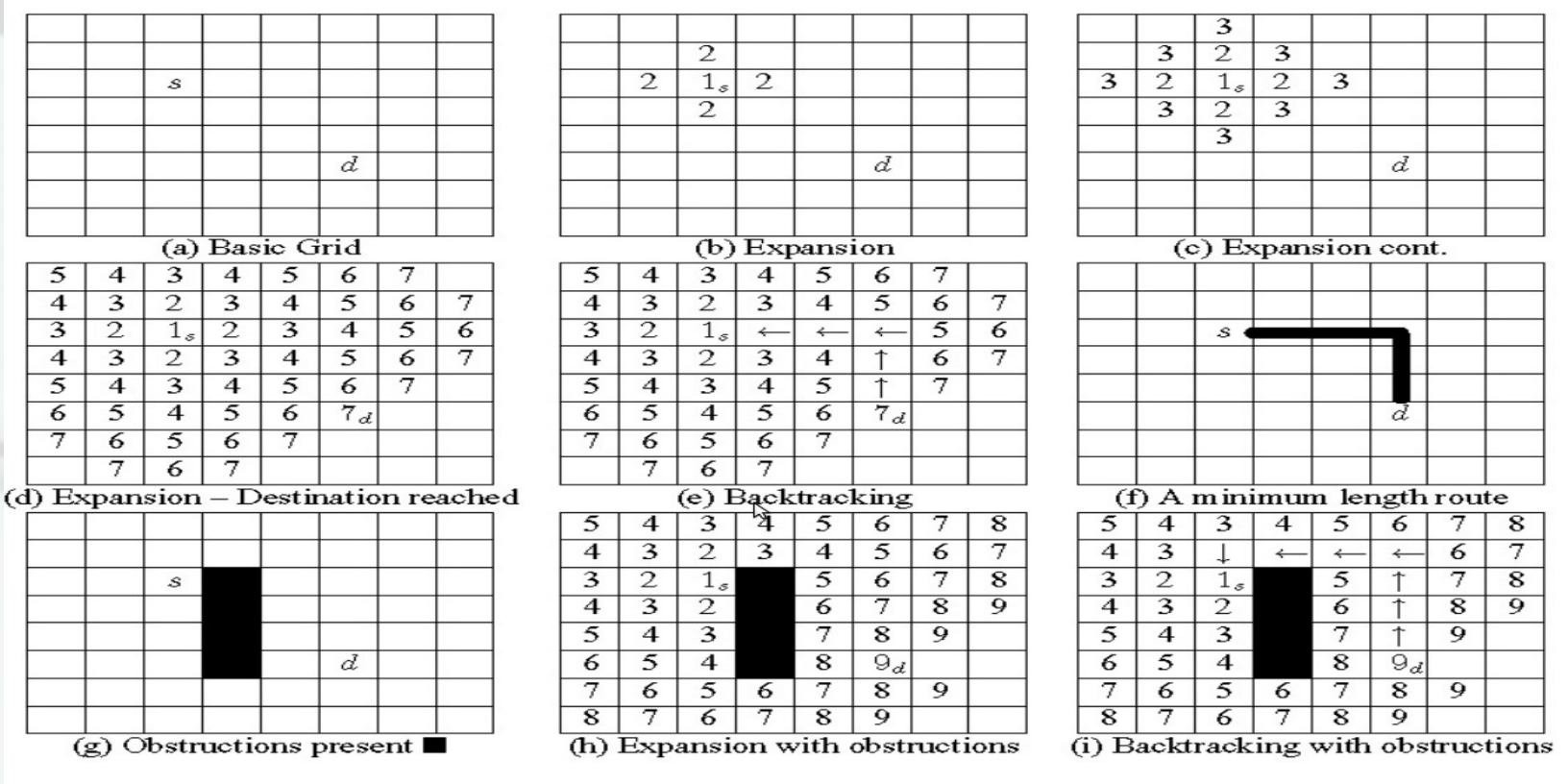
[1]TU Munich [2]INRIA

1.Motivation and Example: LeeTM

- *1.Motivation and Example: LeeTM*
- 2.Approach
- 3.Ongoing and Future Work

Basics on our motivation engine LeeTM

- Each transaction with its own route to connect start point to goal point
- Two phases: expansion and backtracking [fig: Lee-TM PACT2007]



Basics on our motivation engine LeeTM

- Two phases: expansion and backtracking

```
Grid global;
forall routes{
        atomic {
            Expand from the source to the destination;
                  //reads and writes to the global
            Backtrack from destination to source;
                 // reads and writes to the global
           Reset Expansion;
   }
}
```

Control-Flow between Expansion and Backtracking phases

```
bool Lee::connect(WorkQueue *q, int*** tempq){
        bool succes = true;
        //call the expansion method to return found/not found boolean
        bool found = expandFromTo(xs, ys, xg, yg, maxTrackLen*5, tempq);
        if(found){
                success = backTrackFrom(xq, yq, xs, ys, netNo, tempq);
                //call the backtrack
                if(succes && Lee::VERIFY){
                        eddTrackForVerificaiton(q);
                }
         }
         else {failures++;}
         return success;}
```

2.Approach

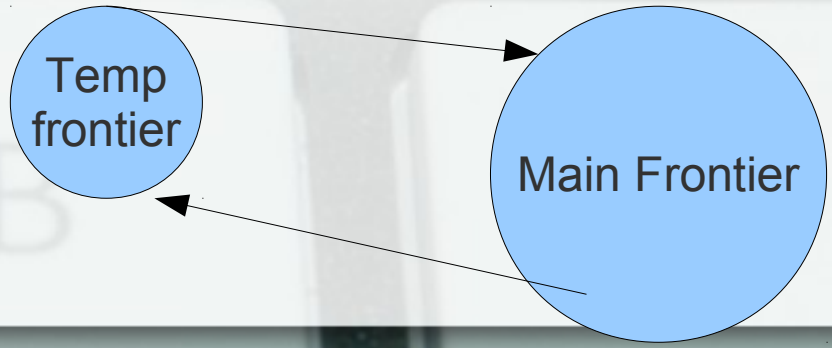
- 1.Motivation and Example: LeeTM
- *2.Approach*
- 3.Ongoing and Future Work

Motivations

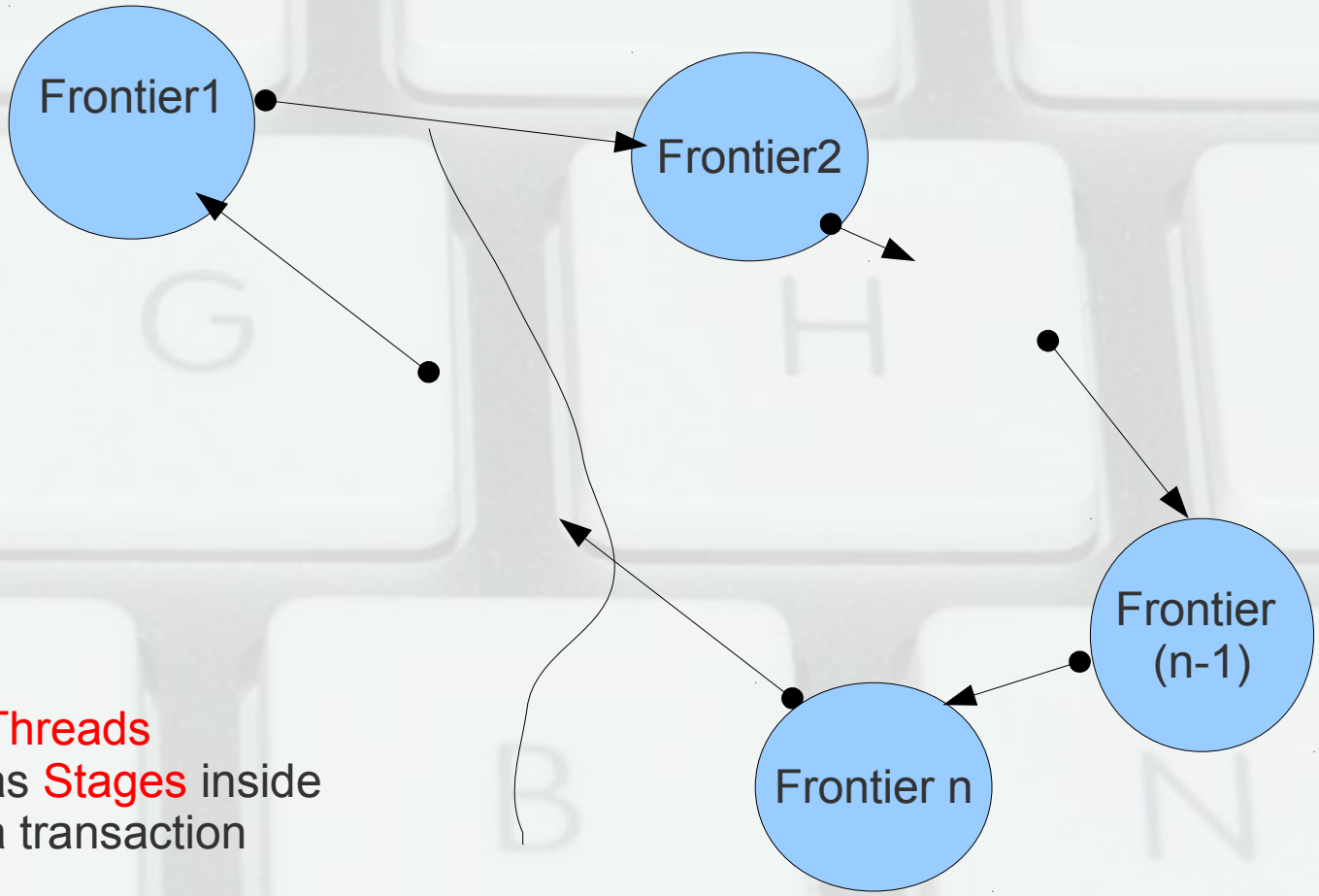
- Motivation 1: Express producer-consumer relation, pipelines
- Motivation 2: Pipeline parallelism inside Transactions

Motivation Point for catching Pipeline inside LeeTM: Expansion Phase

```
while (!front.empty()) {

        while (!front.empty()) {

        .....

        //looking directions, writing to the tempg

        tempg[f.getX() + 1][f.getY()][f.getZ()] = tempg[f.getX()][f.getY()]
           [f.getZ()] + weight; //looking east

        ....

        }

        //swapping between temp front and front

        //between inner and outer loop

        vector<Frontier> tf;

        tf = front;

        front = tmp_front;

        tmp_front = tf;

        }
```



Creating Pipelines inside Expansion



Threads
as Stages inside
a transaction

How the code looks like

```
while(...)
        ...
        while(...)
                {
                    #pragma omp task input(frontier1) output(frontier2)
                            ... = frontier1
                           frontier2 = ...
                }
        while(...)
                {
                #pragma omp task input(frontier2) output(frontier3)
                            ...  = frontier2
                           frontier3 = ...
                }
                ...
        }
```

3.Ongoing and Future Work

- 1.Motivation and Example: LeeTM
- 2.Approach
- *3.Ongoing and Future Work*

Ongoing and Future work

- Trying to find more pipelines inside the TM-Benchmarks
- Integration of Transactional Memory to a data-flow, streaming extension of OpenMP
- Google Summer of Code project titled “Integration of transactional memory support trans-mem GCC into a data-flow extension of OpenMP”
- TERAFLUX Project