Fuzzy Application Parallelization using OpenMP

C. Chantrapornchai
Silpakorn University
Thailand

Abstract

- In this work, we study the parallelism in fuzzy systems using openMP and its possibility in embedded platforms.
- Two versions of the parallelization are mentioned: fine-grained and coarse-grained parallelism.
- We found that the coarse-grained approach is more effective due to the overhead of openMP which becomes more visible in the low-speed CPU. Thus, the coarse-grained approach is suggested. Two versions using parallel-for and section are proposed. Two versions give different speedup rate depending on characteristics of the applications and fuzzy parameters. In general, the experiments convey that as the system runs continuously the openMP implementation can achieve a certain speedup, overcoming the openMP overhead by the proposed parallelization schemes.

Motivation

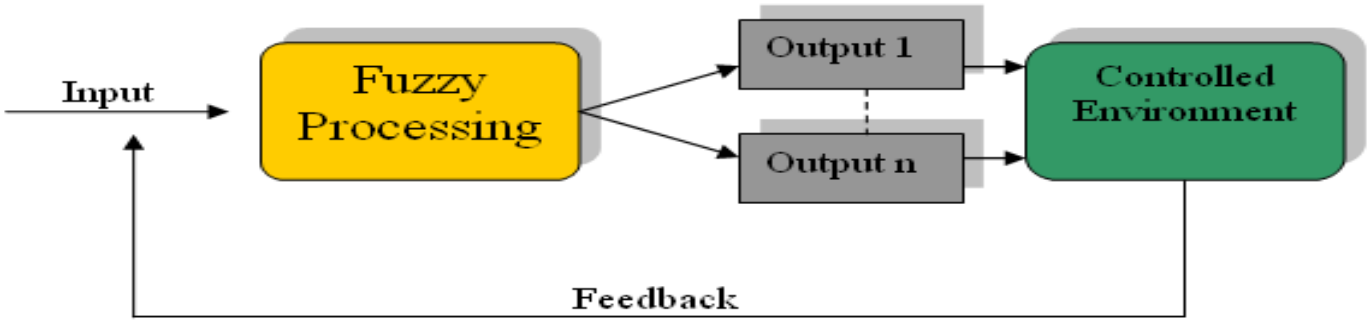
- Fuzzy applications are being used in embedded platforms.
- There are several steps of computations.
- Possible parallelization can be done.
- Existing work are focused on hardware development.

Literatures

- Works on many hardware platform: FPGA, VLSI,Fuzzy processors, microcontrollers, etc.
- Software fuzzy simulation.
- Works on FPGA (compiler) and openMP

Fuzzy computation

- Fuzzification
- Inference
- Defuzzification
- Feedback



Parallelization using openMP

Two choices

- Fine grained
- Coarse grained

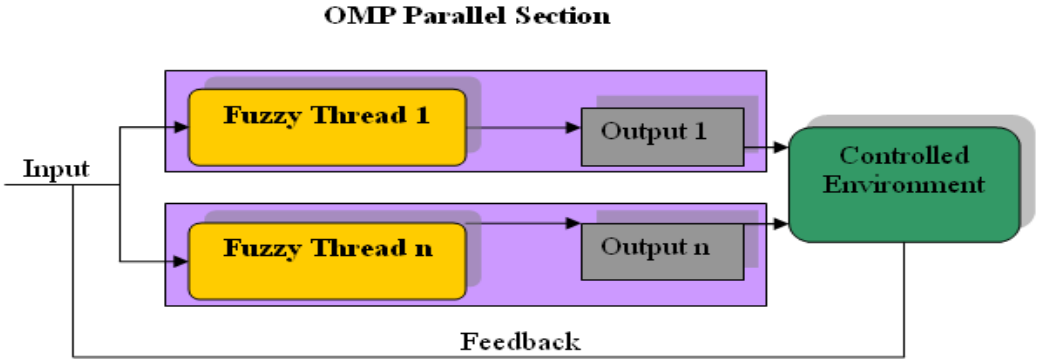
Possible parallel constructs

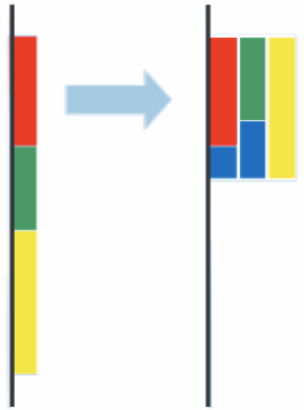
- Parallel for
- Section

Parallelization using openMP

Parallel section

```
#pragma omp parallel sections
{
    #pragma omp section
    {
        .....
    }
    #pragma omp section
    {
        .....
    }
}
```

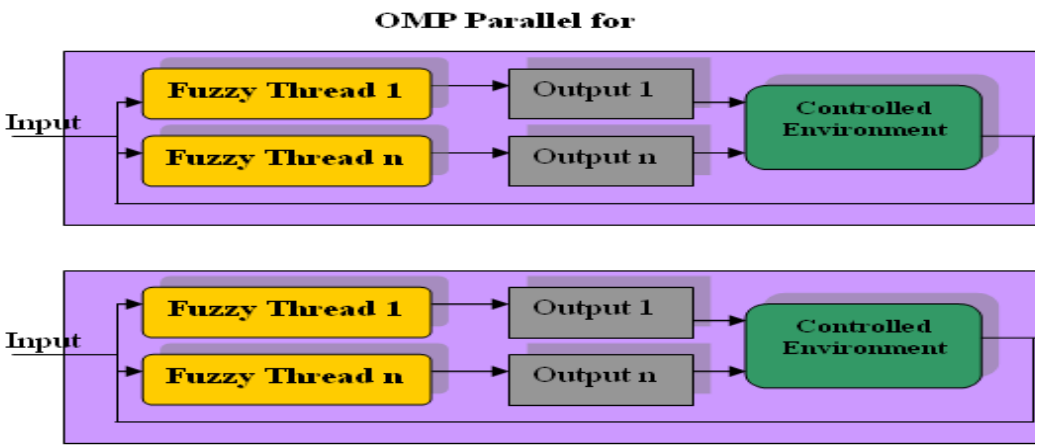


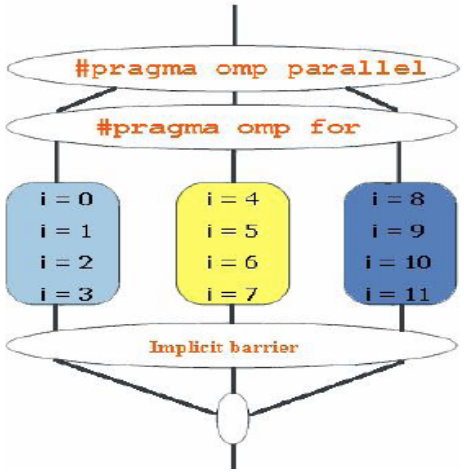


Drawback??

Parallel for

```
#pragma omp parallel
#pragma omp for
for(i=0;i<=n;i++)
{     .....
}
```





Experiments

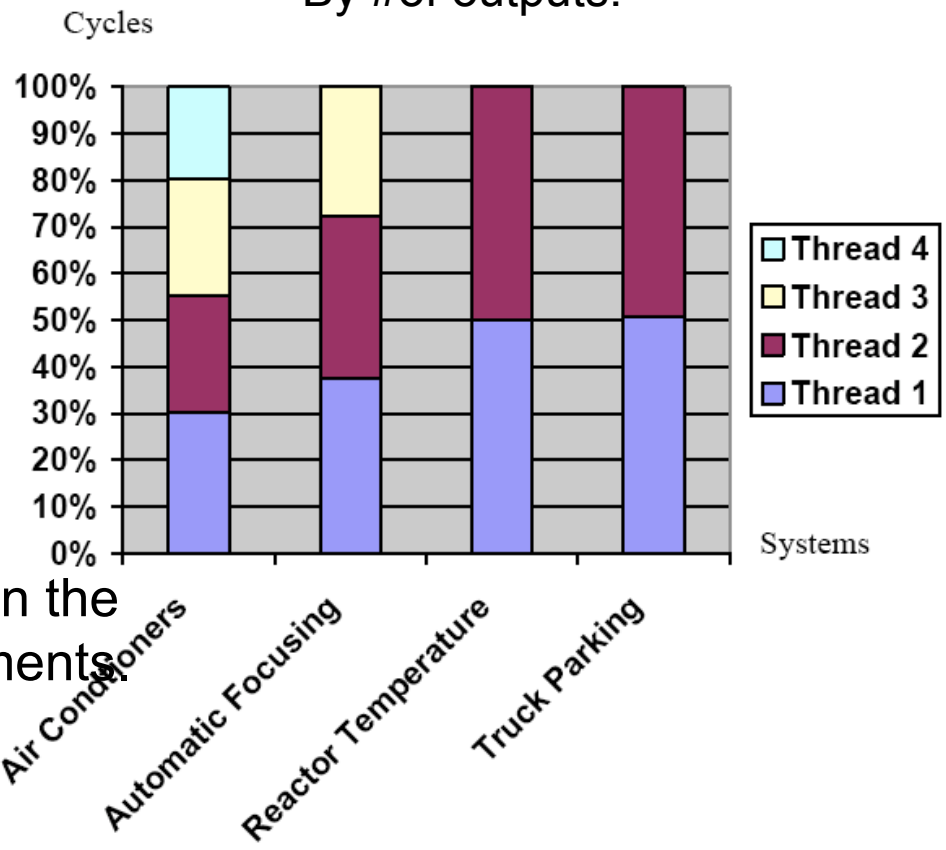
Example	# input	#output	#mem func.	#ele per set	#rules	Infer. method	Defuzz.
1.Air condition[2]	3	4	21	51	96	Max-min	Centroid
2.Automatic focusing system[16]	3	3	51	51	27	Max-min	Centroid
3.Reactor temp. controller [16]	3	2	31	21	254	Max-min	Centroid
4. Truck parking [17]	4	2	27	73	130	Max-prod	Fuzzymean

We measure the number of cycles used for each approach.
The overheads of openMP pragma are also measured. The experiments are run on the machine with Intel Core 2 Quad, Q8200, 2.33GHz, Core 45nm FSB 1333MHz, L2 Cache 4MB,4GB RAM, running WinXP service pack 3. We use Intel OpenMP/C++ compiler.

Experiments

- Workload for each thread

By #of outputs.

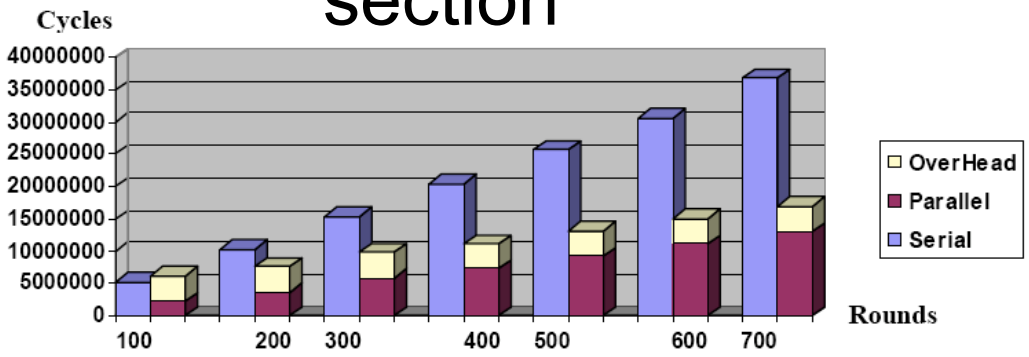


The number of threads are not more than the number of available cores in the experiments.

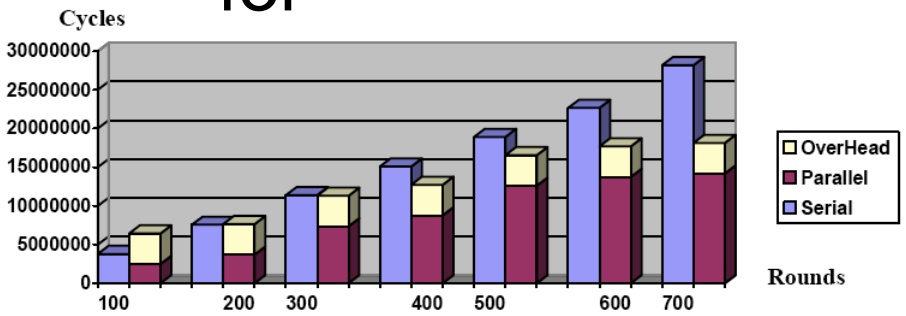
Experiments

- Cycles used

section



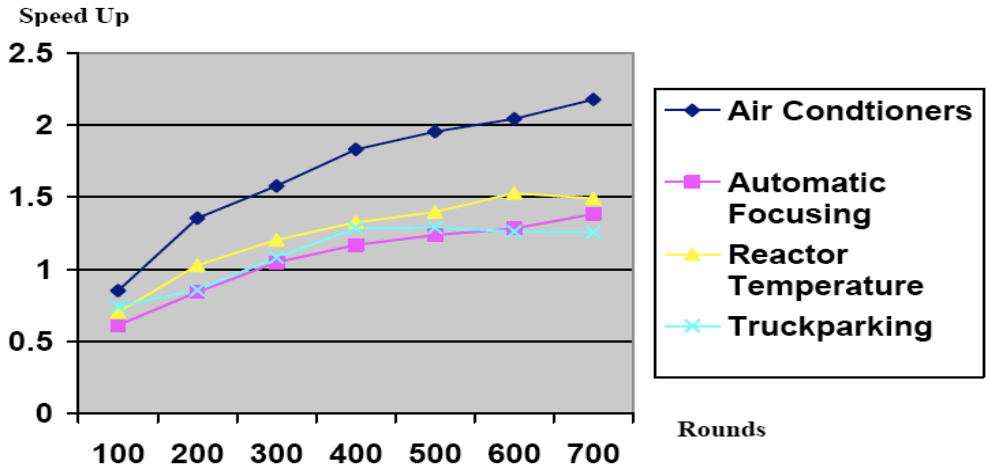
for



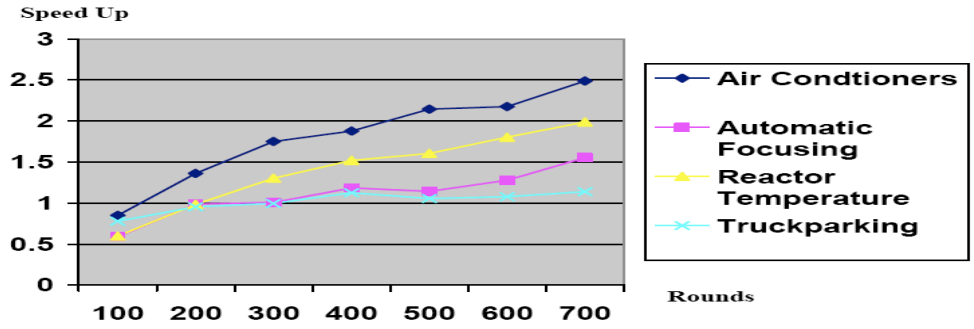
Experiments

- Speedup

section

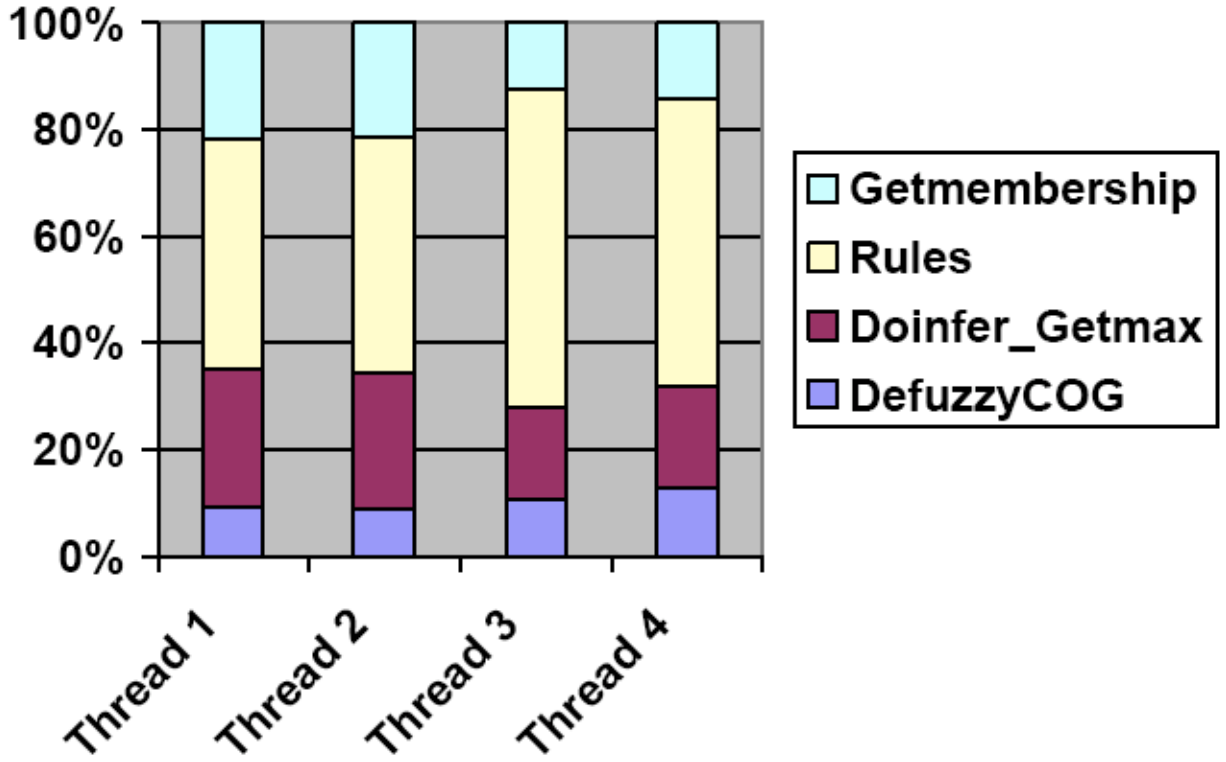


for



Experiments

- Average computation for Air condition



Conclusion

- Coarse grained computation is suggested.
- “For” and “section” are demonstrated.
- “For” has a drawback for dependency but good speedup though.
- For “section”, a thread is mapped to each section.
- Speedup depends on applications.
- Future work will consider other approaches.