Parallel Evolutionary Algorithm in Scheduling Work Packages to Minimize the Duration of Software Project

Jinghui Hu, Xu Wang, Jian Ren, Chao Liu

School of Computer Science and Engineering

Beihang University

NASAC2017@Harbin

Motivations

Software Project Management

1. resources allocation

2. work package scheduling

GPGPU Acceleration

How to accelerate computing by parallelism?

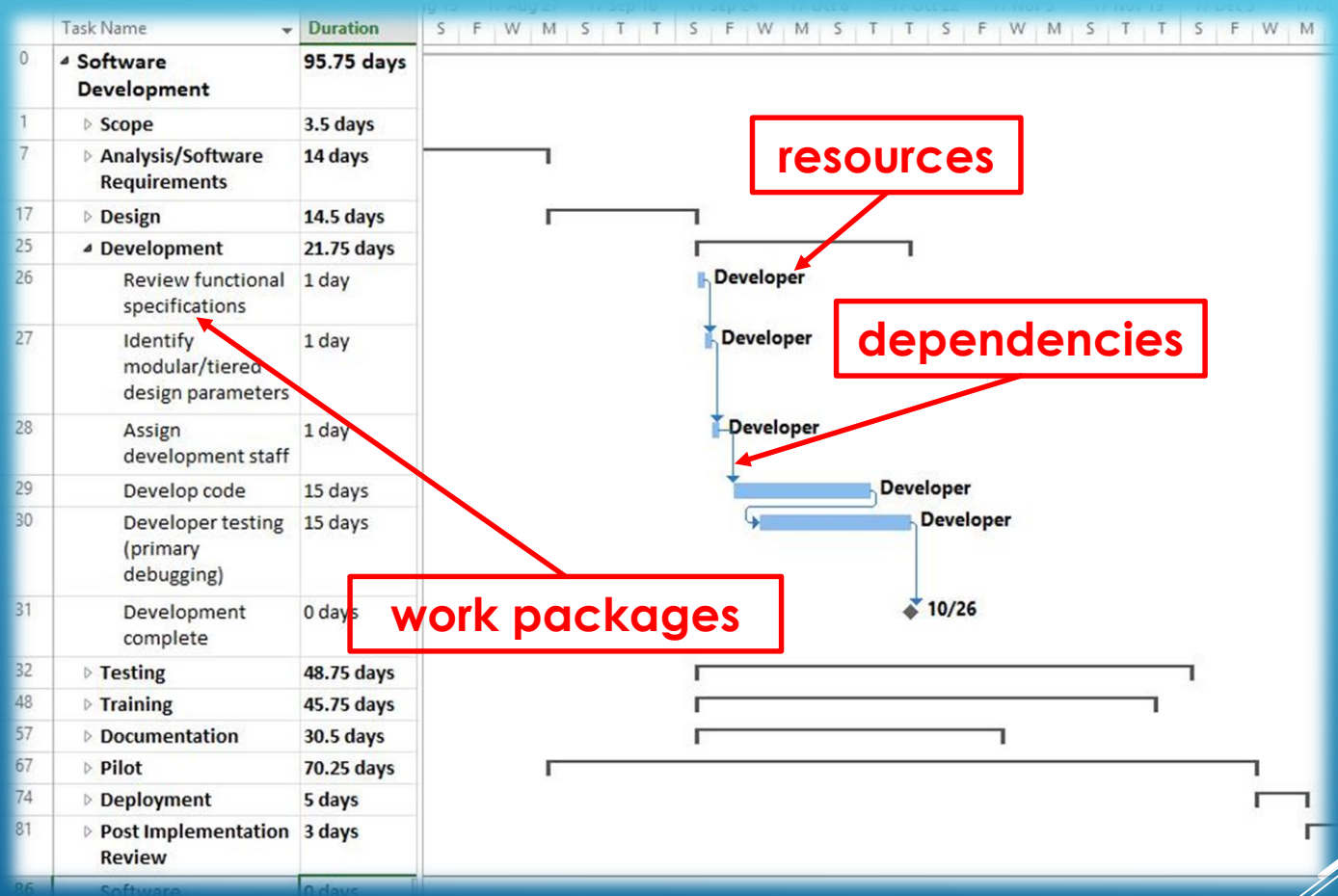
Objective

To find minimal overall duration of a software project by optimizing Work Package Sequence (WPS), while satisfying work package dependencies and resources constraint.

Outline

1. Background & Problem Definition
2. Evolutionary Algorithm
3. Parallelism using GPGPU
4. Research Questions & Results
5. Tool & Future Work

How PM mange a project in Microsoft Project

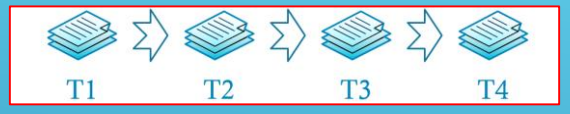


Problem Definition

Description	Notation
Work packages	$T = \{t_1, t_2, \dots, t_N\}$
Resources	$R = \{r_1, r_2, \dots, r_M\}$
Effort (estimated time)	$E = \{e_1, e_2, \dots, e_N\}$
Resources constraint	$\mathrm{TR}(t_i, r_j) = 1$, work package t_i needs r_j; $\mathrm{TR}(t_i, r_j) = 0$, work package t_i does not need r_j;
Dependencies constraint	$Dep = \{t_i \rightarrow t_j \mid t_i, t_j \in T, t_j \ depends \ on \ t_i\}$

The Objective of Optimization

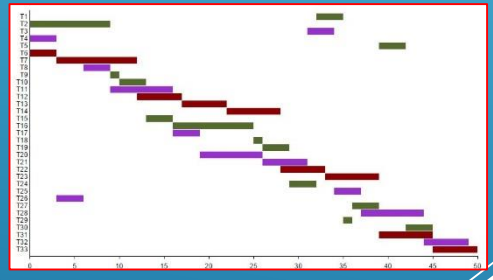
Find an optimal work package sequence:



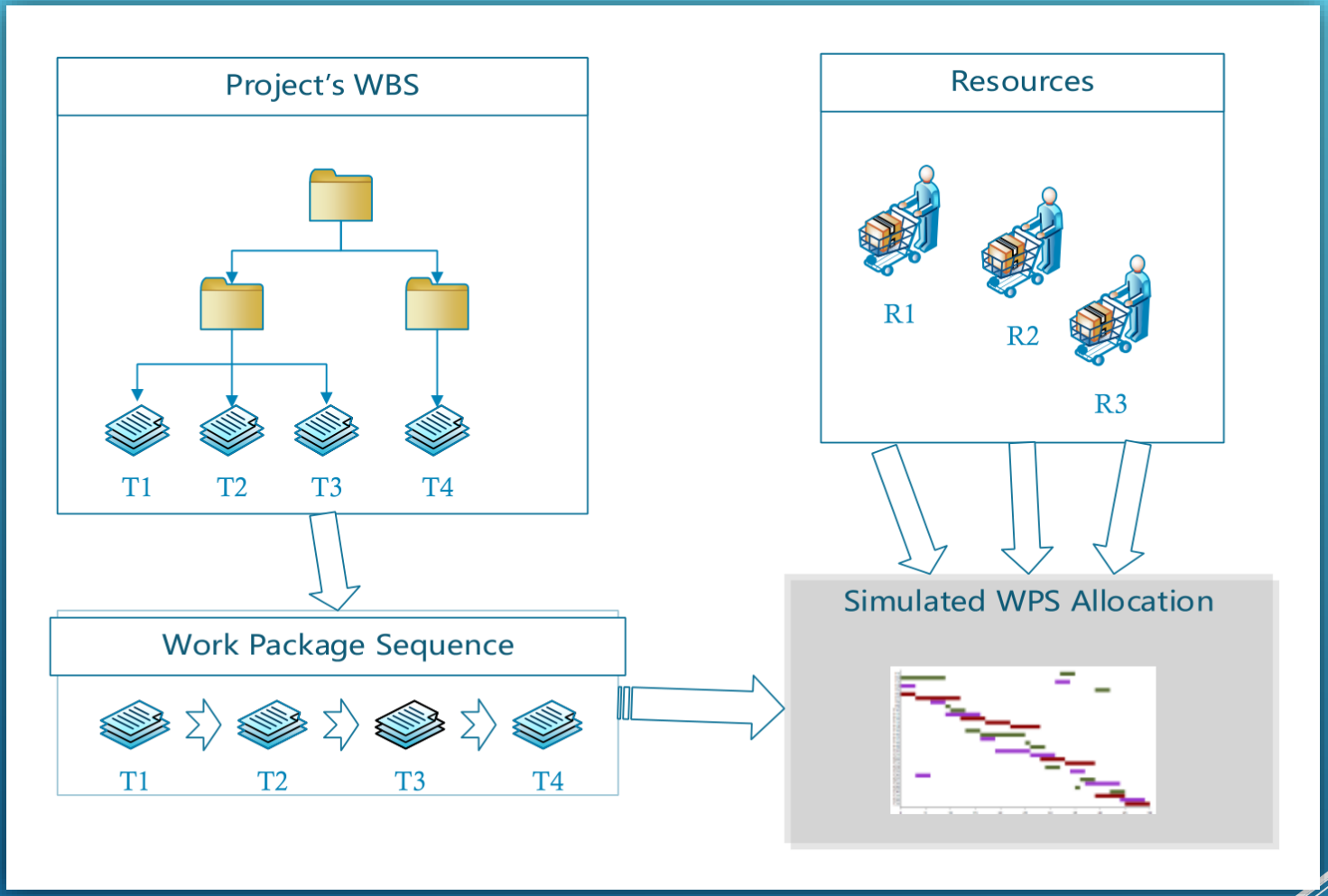
$$S = \{(t_{p_1}, r_{q_1}) \ldots \rightarrow (t_{p_j}, r_{q_j}) \rightarrow \ldots (t_{p_N}, r_{q_N}) \mid t_{p_i} \in T, r_{q_j} \in R\}$$

The Objective is to minimize overall duration:

$$\textbf{Objective: } min(f(S)) = min(max\{t.end \mid t \in T\})$$



How we model the problem



Outline

1. Background & Problem Definition
2. Evolutionary Algorithm
3. Parallelism using GPGPU
4. Research Questions & Results
5. Tool & Future Work

The Representation of Solution

The solution represents the work package sequence

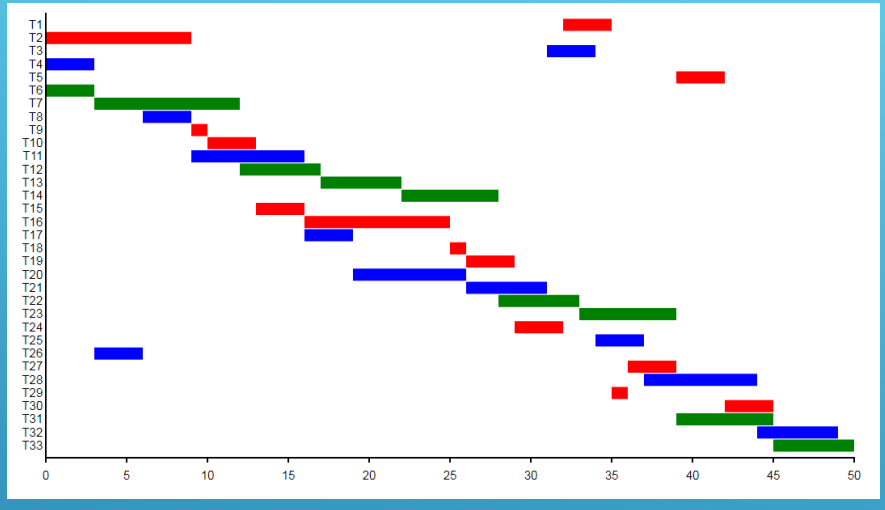
$$T_1 \rightarrow T_5 \rightarrow T_6 \rightarrow T_4 \rightarrow T_8 \rightarrow T_3 \rightarrow T_9 \rightarrow T_2 \rightarrow T_7$$

The work packages are sorted by their priorities as shown above.

Size of the solution space:
The total number of all possible solutions for a project containing N work packages is $N!$

Fitness Evaluation

```
input: WPS, R, TE, TR
output: duration
  for i from 1 to WPS.length do
    t ← WPS[i]
    effort ← TE(i)
    for r in R do
      r.occupy ← 0
    end for
    for r in R do
      if TR(t, r) = true then
        t.start ← r.occupy
        t.end ← r.occupy + effort
        r.occupy ← t.end
        break
      end if
    end for
  end for
  duration ← 0
  for t in WPS do
    if t.end > duration then
      duration ← t.end
    end if
  end for
```



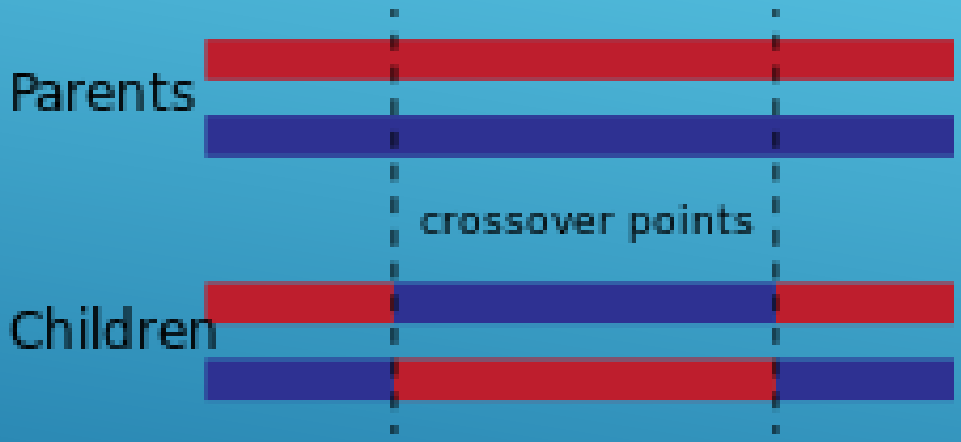
The same color shares the same resource.

Total 33 work packages with 50 days overall duration

http://jeanhwea.github.io/apps/gantt/index.html

Genetic Operators for GA

Crossover: Two-point crossover



Mutation: Two-point exchange mutation

Randomly select two work packages in parents, and exchange the two points to get children.

Outline

1. Background & Problem Definition
2. Evolutionary Algorithm
3. Parallelism using GPGPU
4. Research Questions & Results
5. Tool & Future Work

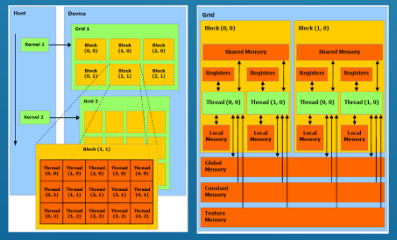
Why Parallelization???

Genetic Algorithm is computational expensive,

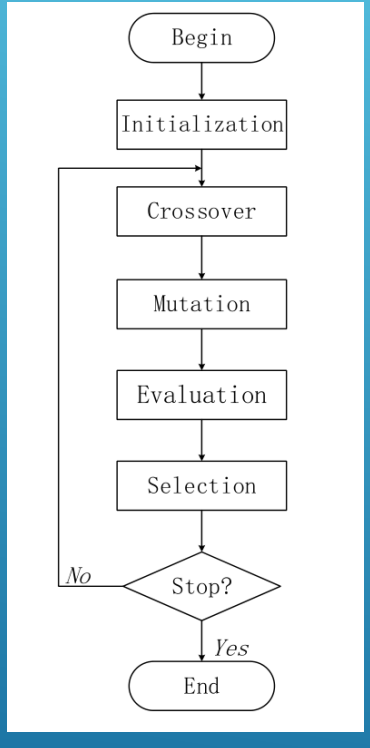
Mangers just don't like to wait hours for an optimized solution.

However, due to the nature of GA, the fitness evaluation of each individual solution is independent from all other solutions in term of data and computation.

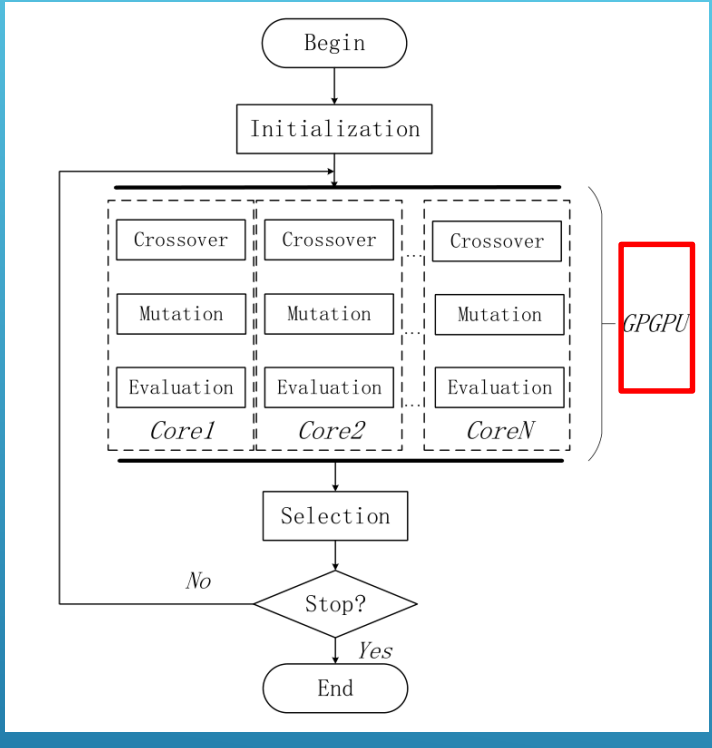
GPU has more cores than CPU!!!



Our Implementation of Parallel Algorithm



sequential



parallel

Sequential v.s. Parallel

```
void gaCrossover(int * dad, int * mom, int * bro, int * sis)
{
    size_t i, j, k, a, b;
    int * dad_new, * mom_new;
    if (genRandProb() < ga_prob_crossover) {
        a = genRandInt(0, sz_task-1);
```

```
void gaMutation(int * person)
{
    size_t a, b;

    if (genRandProb() < ga_prob_mutation) {
        a = genRandInt(0, sz_task-1);
        b = genRandInt(0, sz_task-1);
        if (a > b) {
            swap(a, b);
        }

        swapBits(a, b, person);
```

```
__device__ void gaCrossover(int * dad, int * mom, int * bro, int * sis)
{
    size_t i, j, k, a, b;
    int dad_new[MAX_
    a = randInt(0, 
    b = randInt(0, 
    if (a > b) {
        size_t tmp;
        tmp=a; a=b;
    }
```

```
__device__ void gaMutation(int * person)
{
    size_t a, b;

    a = randInt(0, d_ntask-1);
    b = randInt(0, d_ntask-1);
    if (a > b) {
        size_t tmp;
        tmp=a; a=b; b=tmp;
    }

    swapBits(a, b, person);
```

```
gaCrossover<<<1, npop/2, msize_occupy>>>(h_chrm, h_hashv, h_fitv);
gaMutation<<<1, npop, msize_occupy>>>(h_chrm, h_hashv, h_fitv);
gaSelection();
gaStatistics(fd_info);
```

npop: population size
Put computation into GPU core

Outline

1. Background & Problem Definition
2. Evolutionary Algorithm
3. Parallelism using GPGPU
4. Research Questions & Results
5. Tool & Future Work

Research Questions

RQ1(effectiveness) : Does the evolutionary algorithm effectively optimize project management problem, and get an optimized solution?

RQ2(efficiency) : Is the parallel evolutionary algorithm able to improve the efficiency in the project management problem?

Industrial Project Data

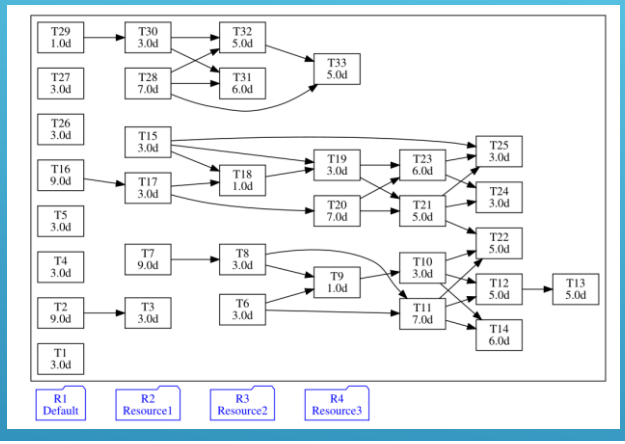
	# work packages	# resources	# dependencies
A-Input	33	4	33
B-DBUpgrade	106	8	105
C-SmartPrice	74	14	73

A-Input: s simulated small-scale project planning.

B-DBUpgrade: upgrade the Oracle database from the 9g version to the 10g version

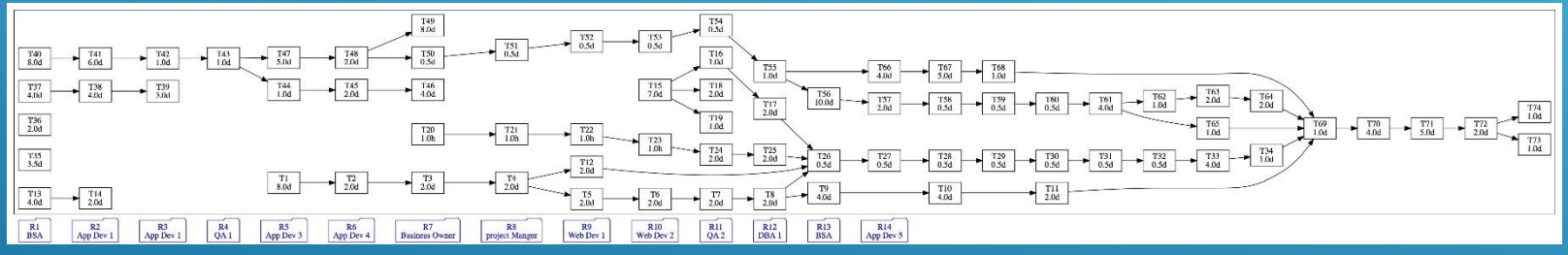
C-SmartPrice: a supply chain enhancement of medium size affecting mostly the website as well as a few internal applications

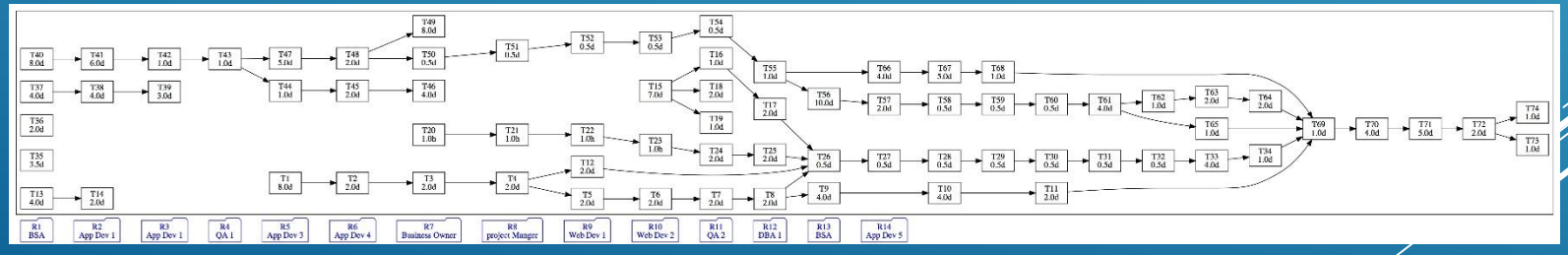
Work Package Dependencies



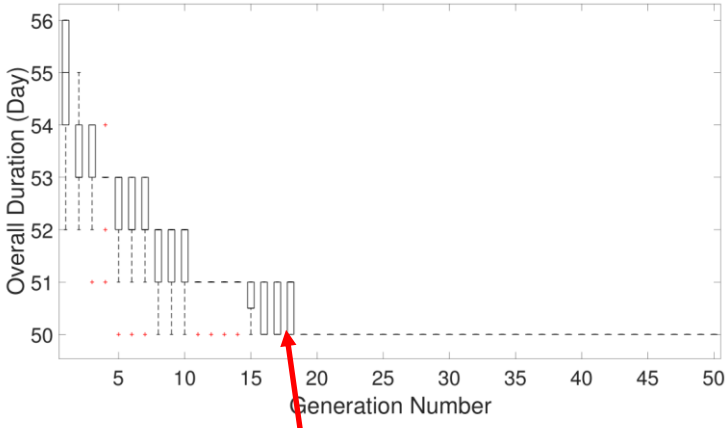
DAG (Directed Acyclic Graph)

1. Black Box -> Work Package
2. Blue Box -> Resource
3. Arrows -> Dependency

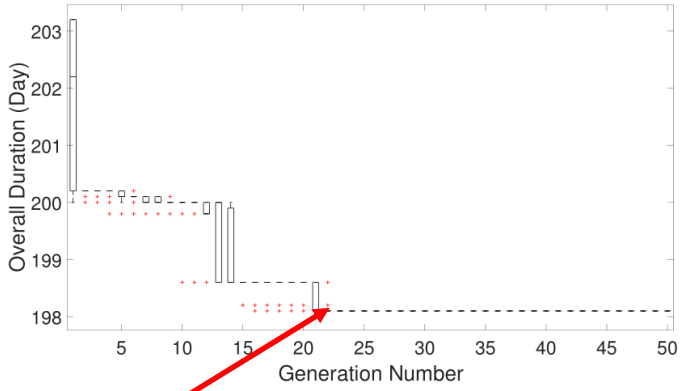




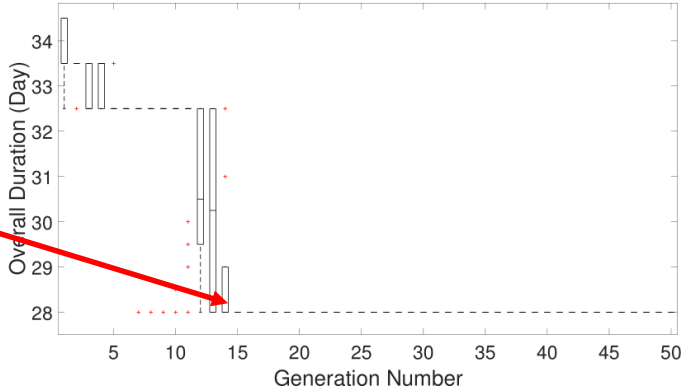
Result for RQ1: Effectiveness



(a) *A-Input*'s solution converges on *19th* generation



(b) *B-DBUpgrade*'s solution converges on *22rd* generation



(c) *C-SmartPrice*'s solution converges on *15th* generation

The solutions converge in optimization progress

Runtime environment of CPU & GPU

CPU	GPU
Intel i7 series	Nvidia GeForce GTX 970
2.93GHz	1.27GHz
8 cores	1664 cores
Visual Studio C++ 2012	CUDA 7.0.28

Result for RQ2: Acceleration

Table 2. The Comparison of Sequential and Parallel Implementation

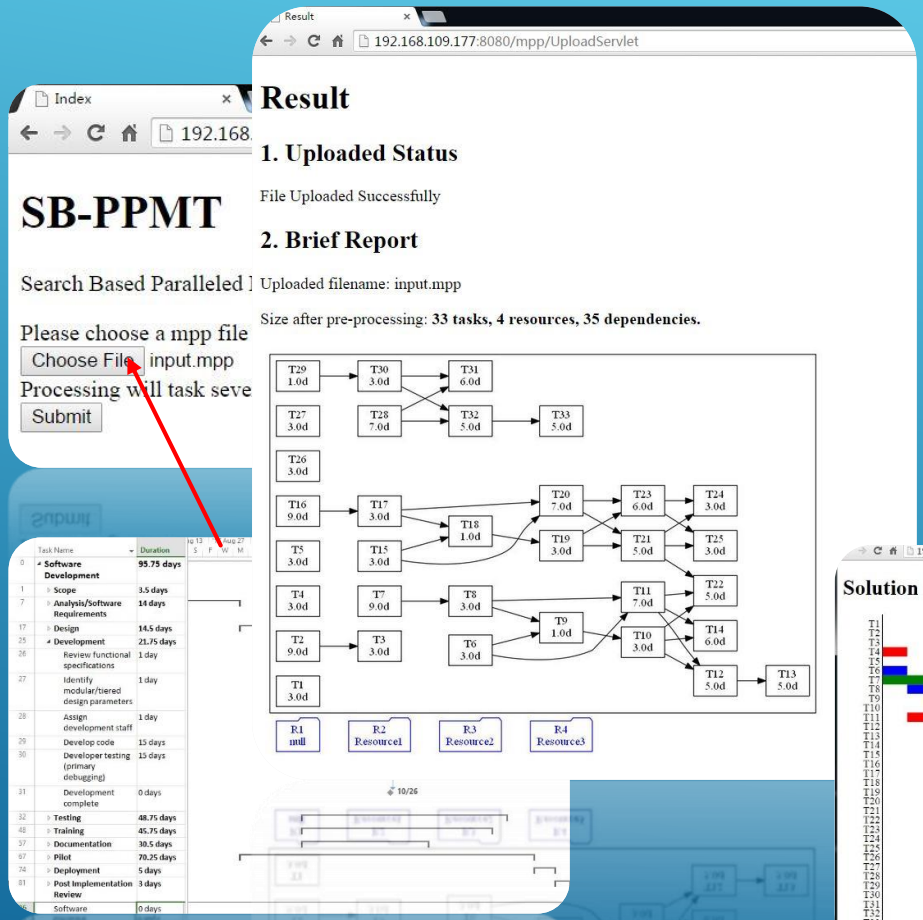
	A-Input		*B-DBUpgrade*		*C-SmartPrice*	
Time(ms)	CPU	GPU	CPU	GPU	CPU	GPU
1	7991.8	4102.5	45225.1	21217.5	29642.9	14844.3
2	8431.8	4279.7	44832.1	21377.9	28733.7	14543.6
3	7504.1	3947.9	44934.9	21227.2	28657.9	15003.4
4	7442.1	4642.5	44197.1	21340.3	29489.5	14882.8
5	7376.9	4263.8	45233.7	22448.2	29379.5	14921.5
6	7385.4	4188.6	45197.5	20956.1	28687.9	14723.3
7	8238.8	4540.2	45305.4	21777.2	29786.3	13978.8
8	7470.7	4226.4	45322.8	21147.1	31231.7	14811.1
9	7604.8	3778.2	45040.7	21004.7	28319.7	14292.0
10	7451.9	4712.4	45970.6	21500.9	28322.1	14429.7
Avg.	7689.8	4268.2	45126.0	21399.7	29225.1	14643.1

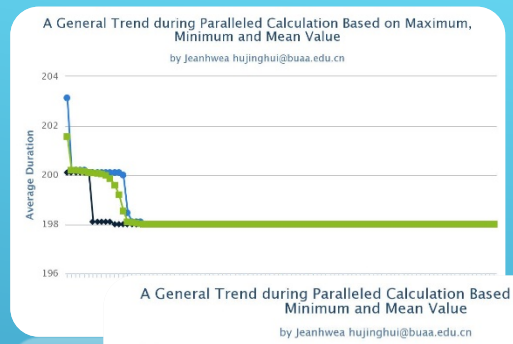
From the table, the executing time of sequential algorithm on CPU is roughly **twice** as long as the parallel algorithm on GPU

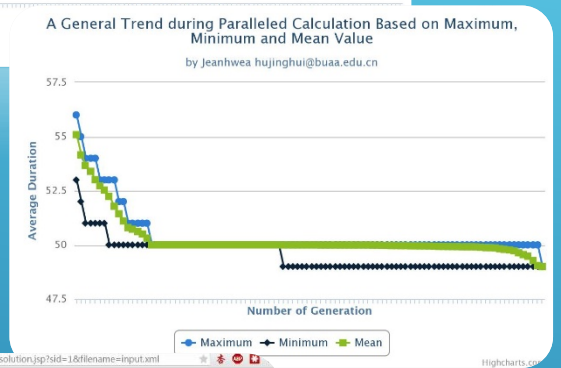
Outline

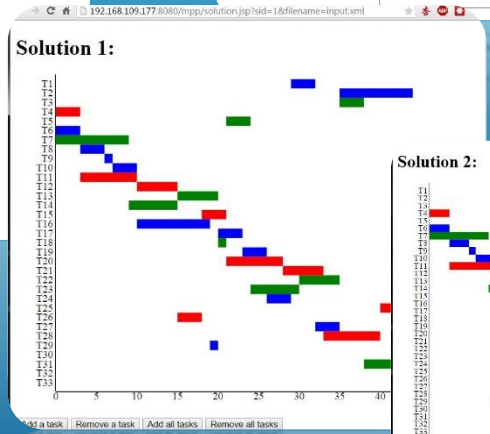
1. Background & Problem Definition
2. Evolutionary Algorithm
3. Parallelism using GPGPU
4. Research Questions & Results
5. Tool & Future Work

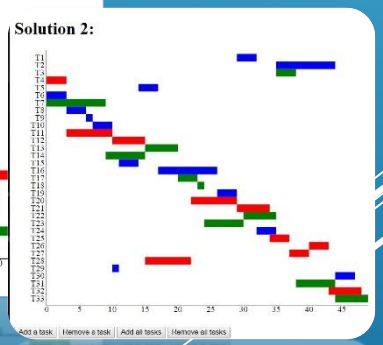
Tool Demo


Index
192.168
SB-PPMT
Search Based Paralleled
Please choose a mpp file
Choose File
input.mpp
Processing will task seve
Submit
Result
192.168.109.177:8080/mpp/UploadServlet
Result
1. Uploaded Status
File Uploaded Successfully
2. Brief Report
Uploaded filename: input.mpp
Size after pre-processing: 33 tasks, 4 resources, 35 dependencies.
R1 mill
R2 Resource1
R3 Resource2
R4 Resource3


A General Trend during Paralleled Calculation Based on Maximum, Minimum and Mean Value
by Jeanhwea hujinghui@buaa.edu.cn
Average Duration
204
202
200
198
196


A General Trend during Paralleled Calculation Based on Maximum, Minimum and Mean Value
by Jeanhwea hujinghui@buaa.edu.cn
Average Duration
57.5
55
52.5
50
47.5
Number of Generation
Maximum
Minimum
Mean


Solution 1:
Add a task
Remove a task
Add all tasks
Remove all tasks


Solution 2:

Future Work

1. more sophisticated problem model

   Staff absence, team construction, etc.

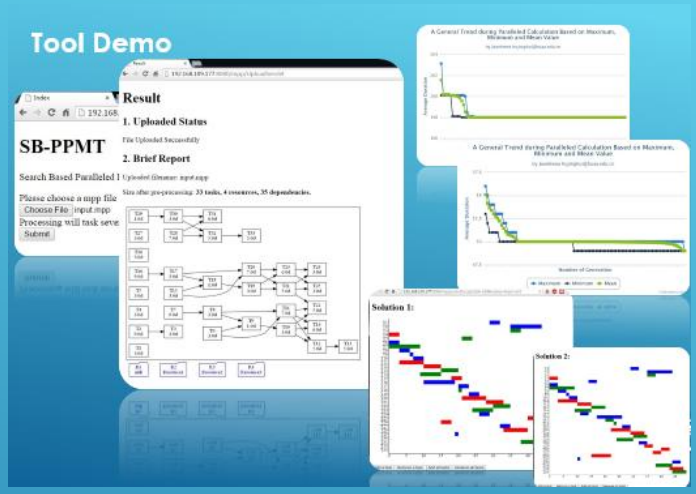
2. Advanced optimization techniques

   Multiple objective, coevolution, etc.

3. Automated decision support

Reviewer's comments

Reviewer 1	Reviewer 2
Q1: Software Project Management Problem is too general. I suggest to clarify the specific problem that is solved in this paper in the title and throughout the paper. ANS: to specific our problem, we change title from "Parallel Evolutionary Algorithm in Solving Single Objective Software Project Management Problem" to "Parallel Evolutionary Algorithm in Scheduling Work Packages and Minimizing Duration of Software Project Management" **Q2**: Using GPGPU and parallel programming to optimize algorithms for software engineering problems is not new idea. The paper should compare the work presented in this paper with other similar works. ANS: We have added some related research paper, such as solving some familiar software management problems with ant colony algorithm, and also added some general idea in GPU acceleration techniques. **Q3**: The optimization of evolutionary algorithm actually is independent from specific problems (e.g., the project management problems addressed in this paper). More explanations are required to better clarify why the optimization of evolutionary algorithm should be explored under the specific setting of software project management problems. ANS: added following discussion in chapter 2 "evolutionary algorithm":"As is known to all, the difficulty of software project management problem is to find the right order of the work package under the precondition of satisfying the constraint of the resource. However, there are numberless constitute of the work package. Traversal is the easiest approach but it requires too much computing resources and results to lower effectiveness. So we need an approach which consumes less time with a perfect solution."	**Q1**: In my opinion, the software management optimization is not only studied in software engineering community, but also in other communities like applied soft computing, operation research. I think the current related work section is not adequate. Actually, there are plenty of studies on software project planning or scheduling optimization, maybe the authors can consider citing more papers, such as: … ANS: This two papers are very close to my paper's topic, The first paper introduced a COCOMO II model to describe sofrware project portfolios and a multiobjective evolutionary approach, mPOEMS. The second introduce an ant colony optimization approach which is called ACS-SPSP algorithm. We have added those works to our Related works. first: "In 2011, Thomas proposed an approach to describe software project problems with a set of multi-objective criteria for portfolio managers using the COCOMO II model and introduce a multi-objective evolutionary approach." second: "In 2013, Jing began solving software project scheduling problems with ant colony optimization." **Q2**: The parallel EA is not a brand-new technique. Even in SBSE, Guo et al. have successfully applied parallel boosted GA to solve a similar SESE problem --- the optimal feature selection problem in software product line. Hence, the technical contribution or innovation of the paper is not significant. ANS: In this paper, five novel parallel algorithms have been proposed to solve multi-objective combinatorial optimization. The five algorithms are ParGIA, OS-GIA, OS-ParGIA, FS-GIA and FS-ParGIA, which GIA is guided improvement algorithm. so we adding following discussion in our paper's Related works. **Q3**: To the best knowledge of mine, in SBSE, the recent trend is to model the optimization problem as a multi-objective problem. For example, for the project management problem, two objectives can be considered: to minimize the overall duration of the project and to minimize the resources consumed in the project. Currently, the authors just consider the first goal. In future, to make the research more practical, the authors may consider some other goals. ANS: We adding some future goal in future work and we find the more development goal: "as the development of the scale of the software project, there are more than one objectives in the problem define. so we will increase the complexity of the model of the software project management problem and improved our algorithm to solve multi-objective problem." **Q4**: The problem statement can be further improved. It is better to clearly state whether the constraints of the project management optimization (i.e., work package dependency and the resource allocation restriction) are liner or non-linear constraints. Also for the single objective, is it a liner or non-linear fitness function? It is a little unclear for me whether EA is the best solution for this kind of SBSE problem. Maybe the authors can add some discussion. ANS: to let the definition more clear, change "corresponding dependencies" to "directed acyclic dependencies". Those constraint is a reduced to a DAG. to make more clear in fitness function's algorithm complexity, adding some discussion in Chapter Fitness Evaluation. add some discussion on fitness function's complexity.



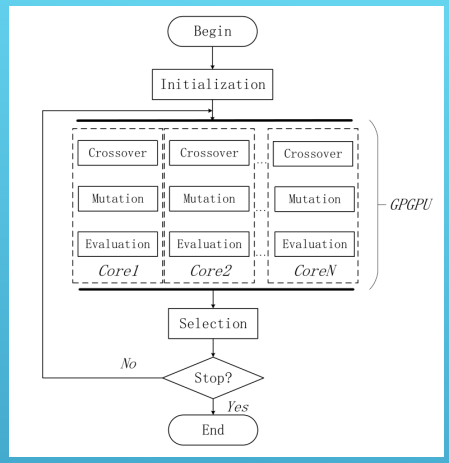
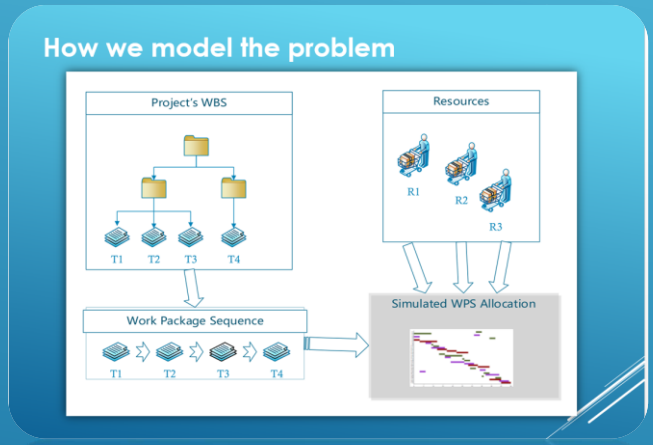


Table 2. The Comparison of Sequen

	A-Input		*B-I*
Time(ms)	CPU	GPU	CPU
1	7991.8	4102.5	45225
2	8431.8	4279.7	44832
3	7504.1	3947.9	44934
4	7442.1	4642.5	44197
5	7376.9	4263.8	4523



Jinghui Hu, Xu Wang, Jian Ren, Chao Liu, Beihang University
{hujinghui, bhwangxu, renjian, liuchao}@buaa.edu.cn