Energy-Efficient Scheduling for Parallel Applications Running on Heterogeneous Clusters

Ziliang Zong

Department of Computer Science and Software Engineering

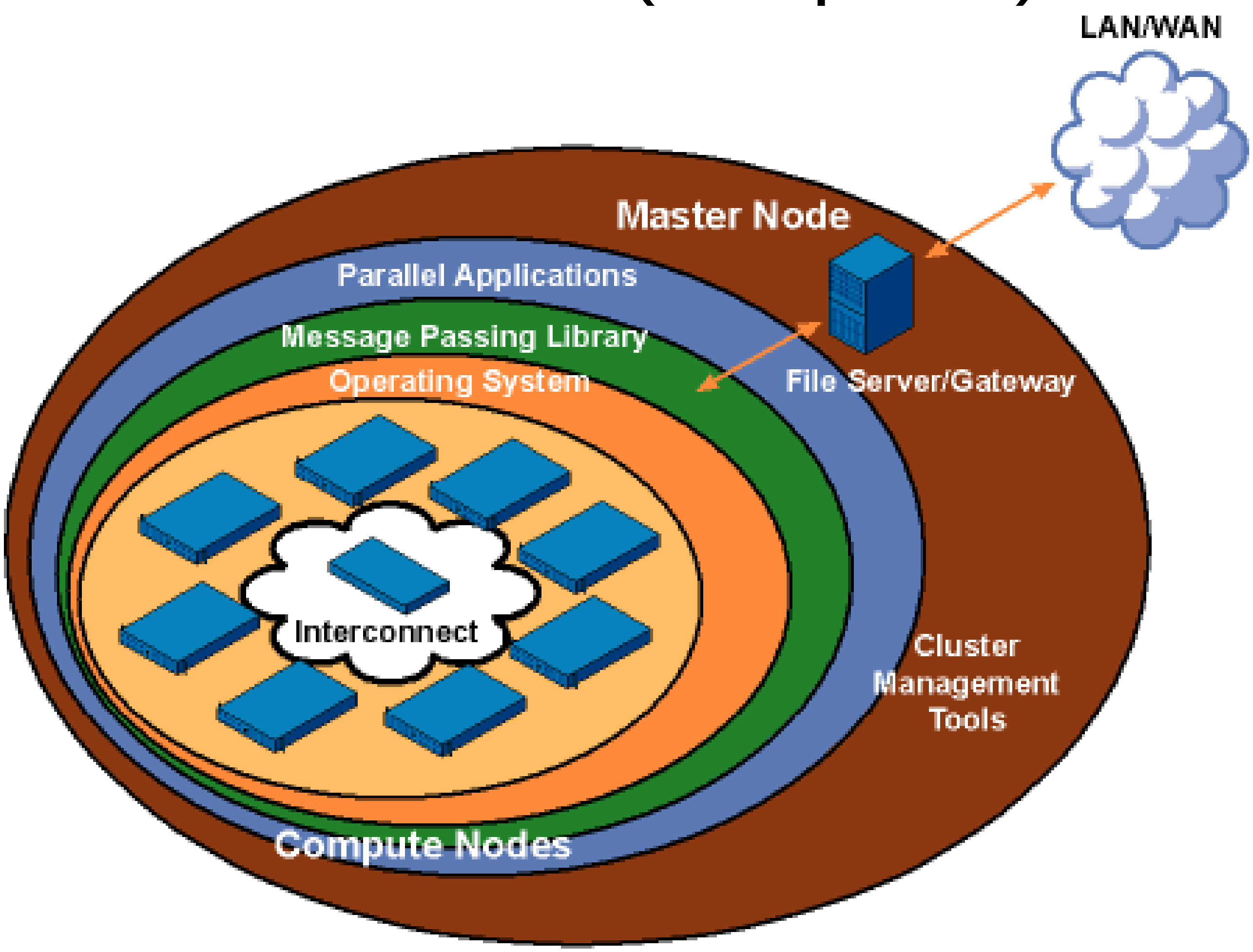
Auburn University

Content

- Introduction
- Related Work
- Energy-Efficient Scheduling
- Preliminary Results

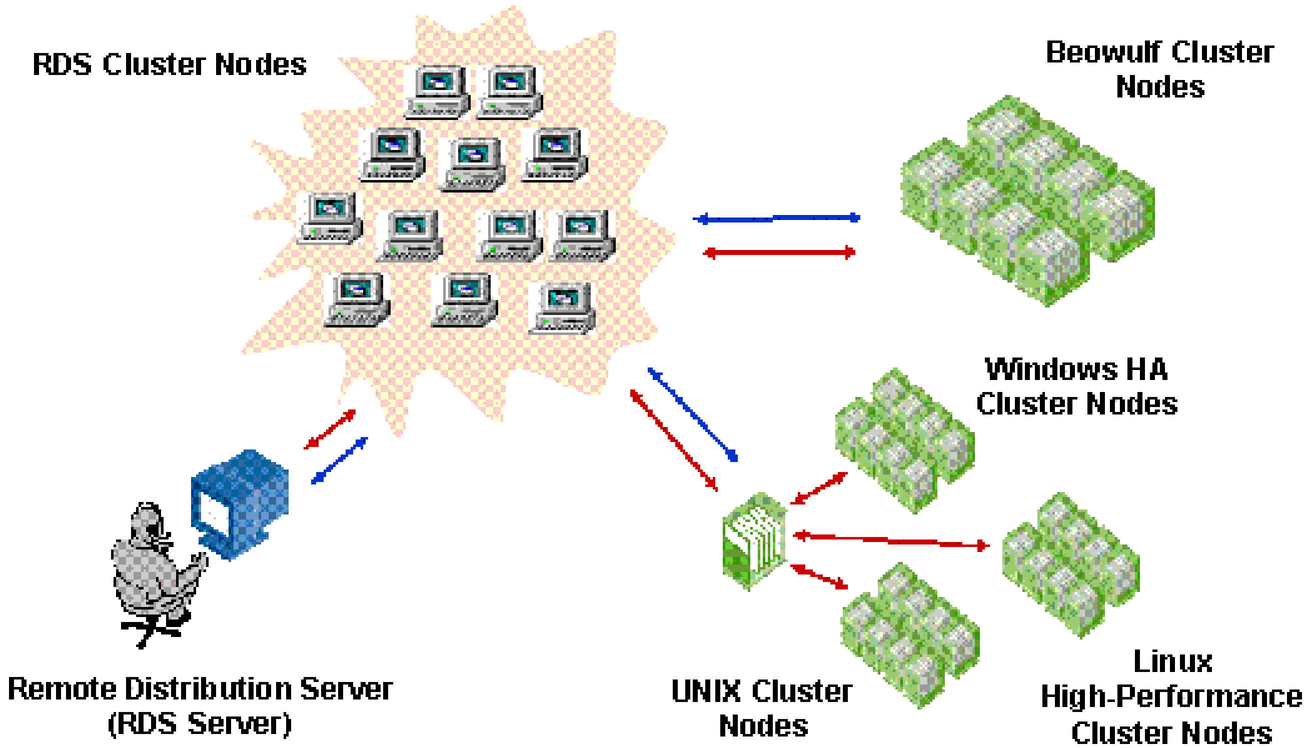
Introduction

- Cluster Architecture (Wikipedia)



Introduction

- Homogeneous & Heterogeneous



Introduction

- Why manage energy usage in clusters is important?
  - **Technical reason**

    Failure rate of computing nodes will increase when cooling system is insufficient because too much heat are dissipated.
  - **Financial reason**

    The new data center capacity projected for 2005 in U.S. would require approximately 40TWh ($4B at $100 per MWh) per year to run 24x7
  - **Environmental reason**

    Generating 1 kWh electricity results in an average 1.55 pounds (lb) of carbon dioxide (CO_2) emissions.

Related Work

- Dynamic Cluster Status Reconfiguration
  Published by Pinheiro et al. and Chase et al. in 2001

- Dynamic Voltage Scaling Policy
  Published by Elnozahy et. al. in 2003

- Request Batching Policy
  Published by E.N. Elnozahy, M. Kistler, and R. Rajamony (IBM Research) in 2003

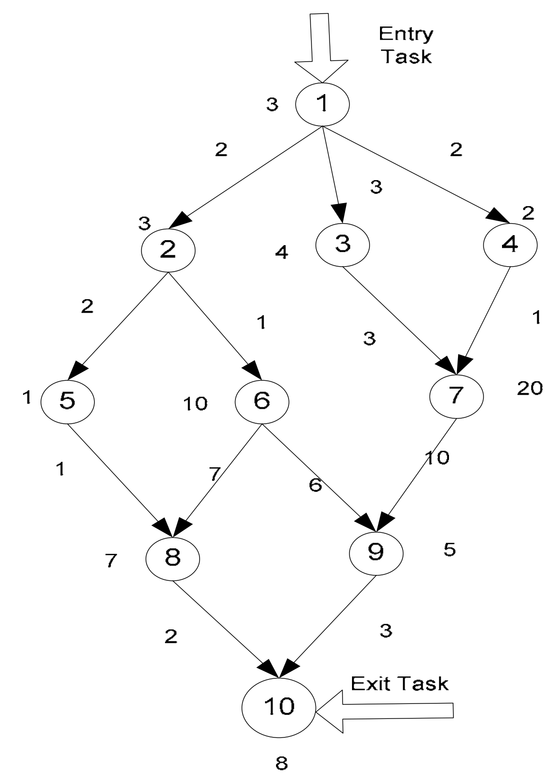
Energy-Efficient Scheduling

- What are the key features of this algorithm?
  - Focus on tasks with dependence
  - Duplication based strategy
  - Consider both CPU and network energy
  - Offline scheduling
  - Heterogeneous environment

Energy-Efficient Scheduling

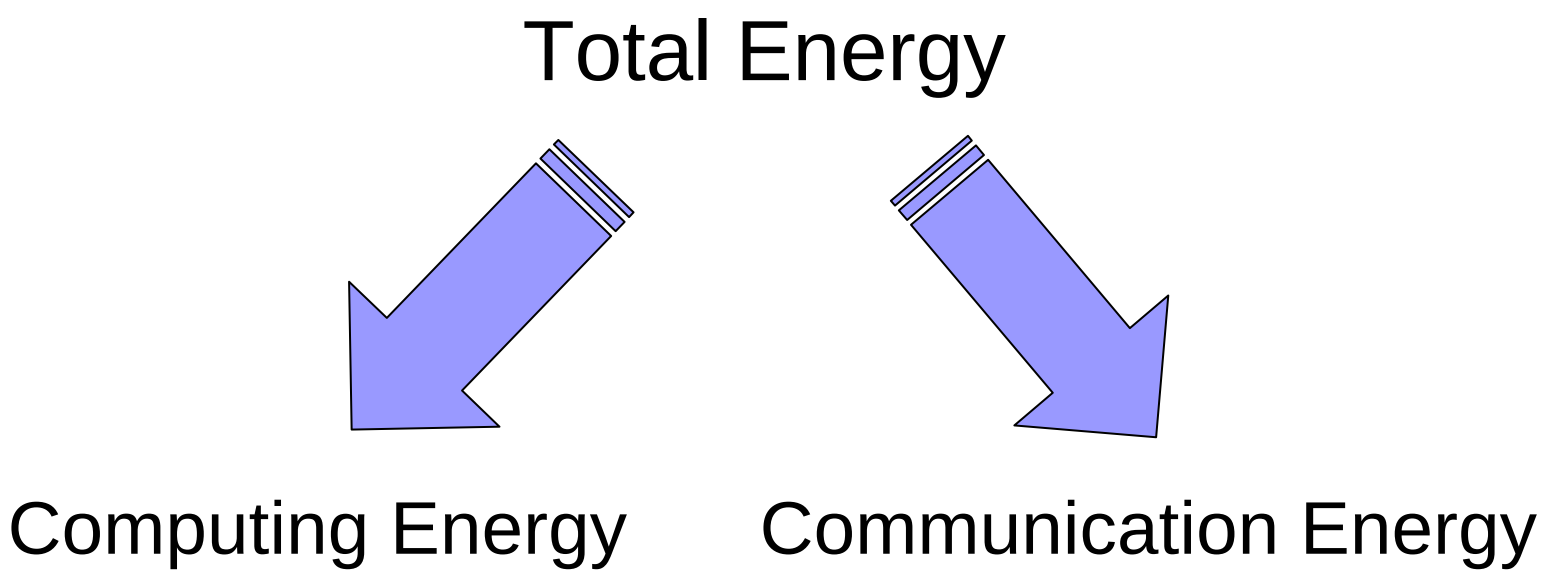
- Model of tasks with dependence

A parallel application with a set of precedence constrained tasks is represented in form of a *Directed Acyclic Graph* (DAG), modeled as a pair *(V, E)*.



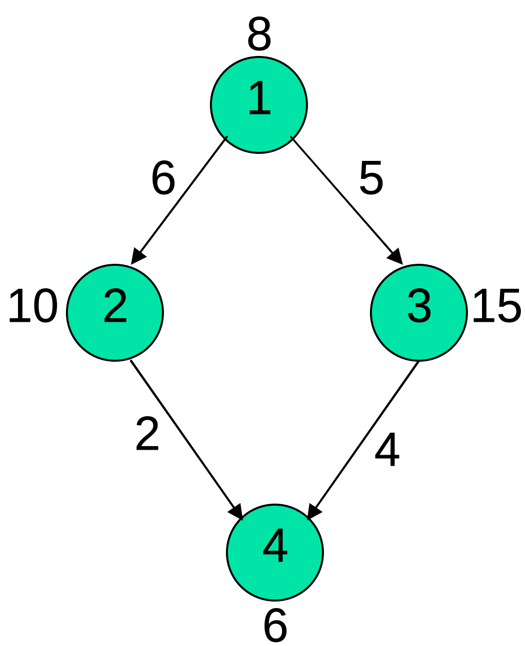
Energy-Efficient Scheduling

- Energy Consumption Model

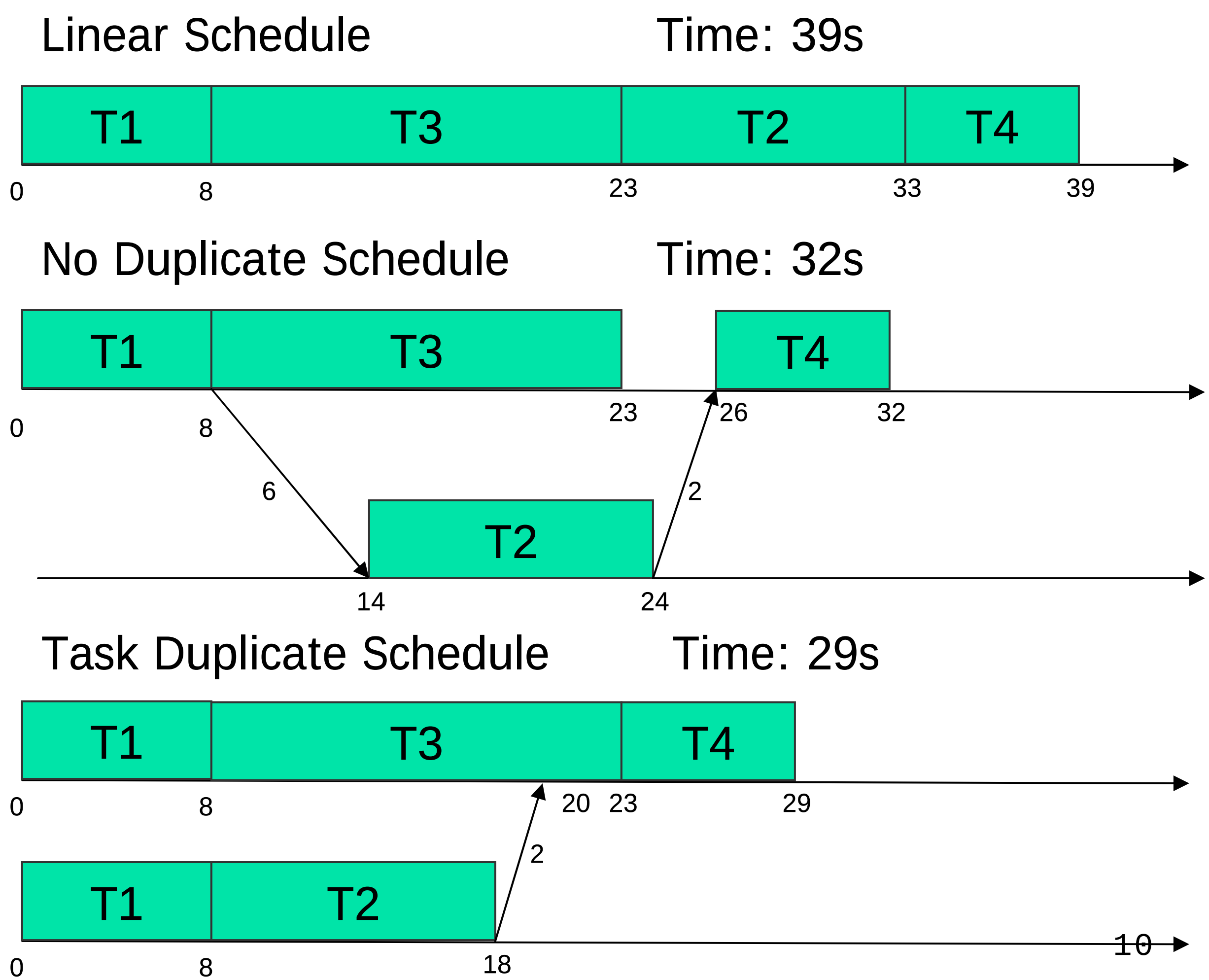


Energy-Efficient Scheduling

- Why duplication for dependent parallel tasks?

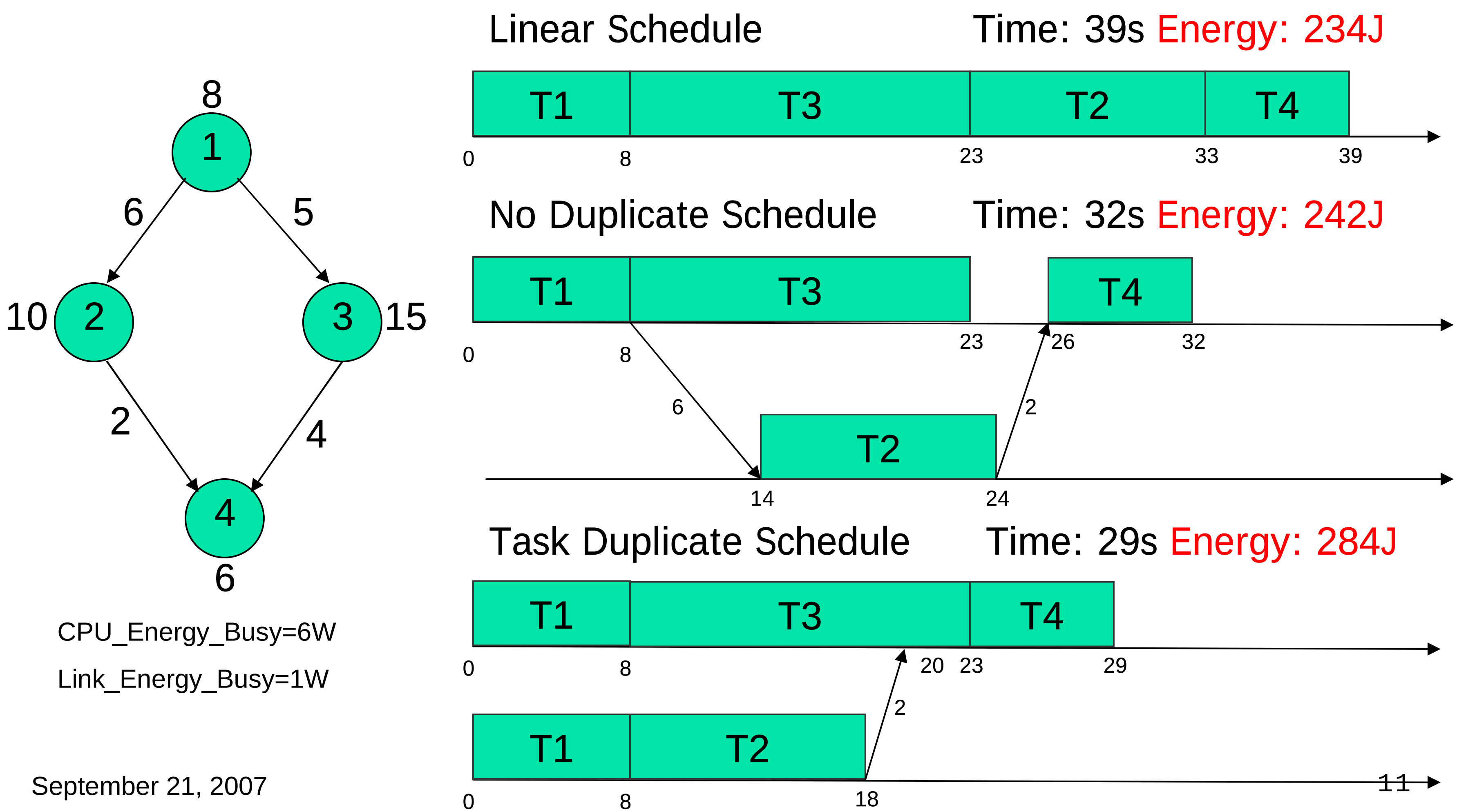


An Example of duplication



Energy-Efficient Scheduling

- Performance is improved. How about energy?



CPU_Energy_Busy=6W

Link_Energy_Busy=1W

CPU_Energy_Busy=6W

Link_Energy_Busy=1W

Here I ignore idle energy

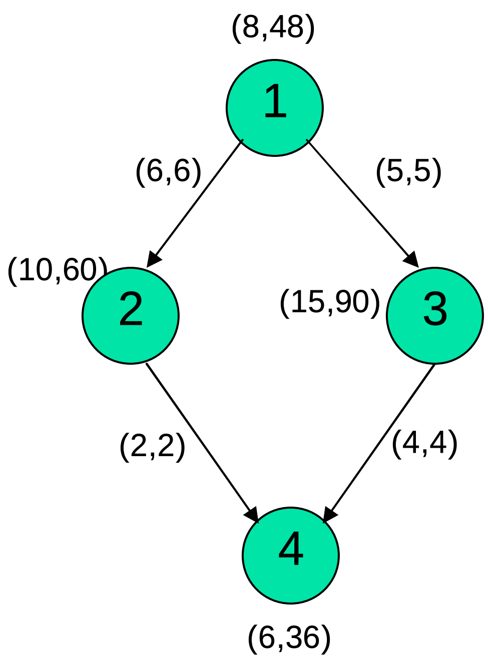
of processors and links

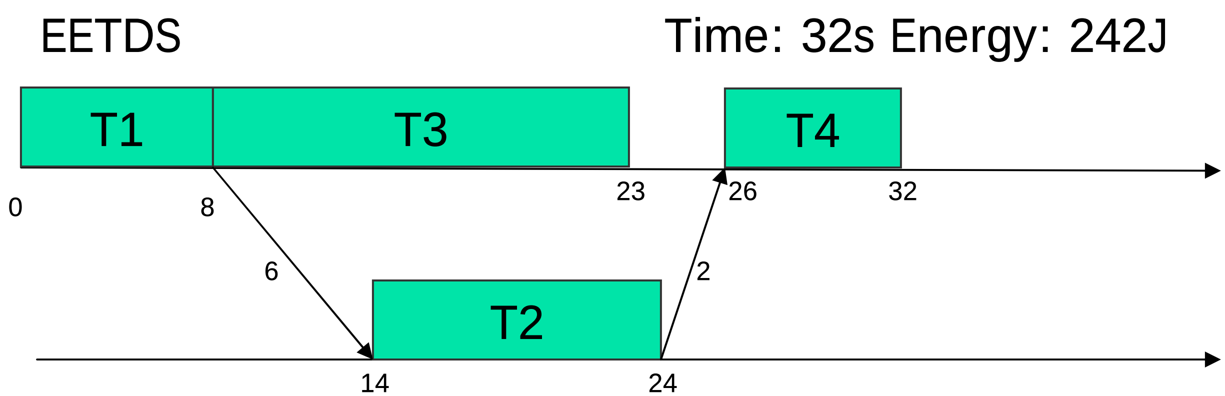
If duplicate T1

More_energy = 48-6 = 42J

If we set threshold = 10 J

EETDS will not duplicate T1





Too much energy consumption

NO Duplication

Energy-Efficient Scheduling

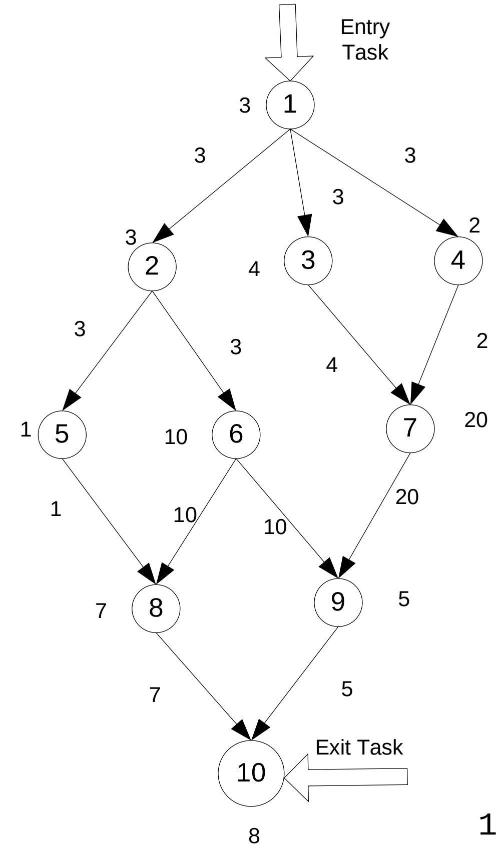
- Implementation (An Example)

Task Description:
TaskSet {T1, T2, …, T9, T10 }
T1 is the entry task;
T10 is the exit task;
T2, T3 and T4 can not start until T1 finished;
T5 and T6 can not start until T2 finished;
T7 can not start until both T3 and T4 finished;
T8 can not start until both T5 and T6 finished;
T9 can not start until both T6 and T7 finished;
T10 can not start until both T8 and T9 finished;

Energy Parameters:
CPU: Intel Core2 Duo E6300
CPU_busy: 44W
CPU_idle: 26W
Merynet_busy:33.6W
Merynet_idle: 8W
Threshold=25J

Time Parameters:
Execution time of each task is given
Communication time between every two tasks is given

Step 1: Generate DAG based on given conditions



Energy-Efficient Scheduling

- Implementation (An Example) Cont.

Step 2: Calculate important parameters

Task	Level	EST	ECT	LAST	LACT	FP
1	40	0	3	0	3	--
2	28	3	6	4	7	1
3	37	3	7	3	7	1
4	35	3	5	3	5	1
5	16	6	7	16	17	2
6	25	6	16	7	17	2
7	33	7	27	7	27	3
8	15	16	23	18	25	6
9	13	27	32	27	32	7
10	8	32	40	32	40	9

Level: Time needed between current task and final completion time

EST: Earliest Start Time

ECT: Earliest Completion Time

LAST: Latest Allowable Start Time

LACT: Latest Allowable Completion Time

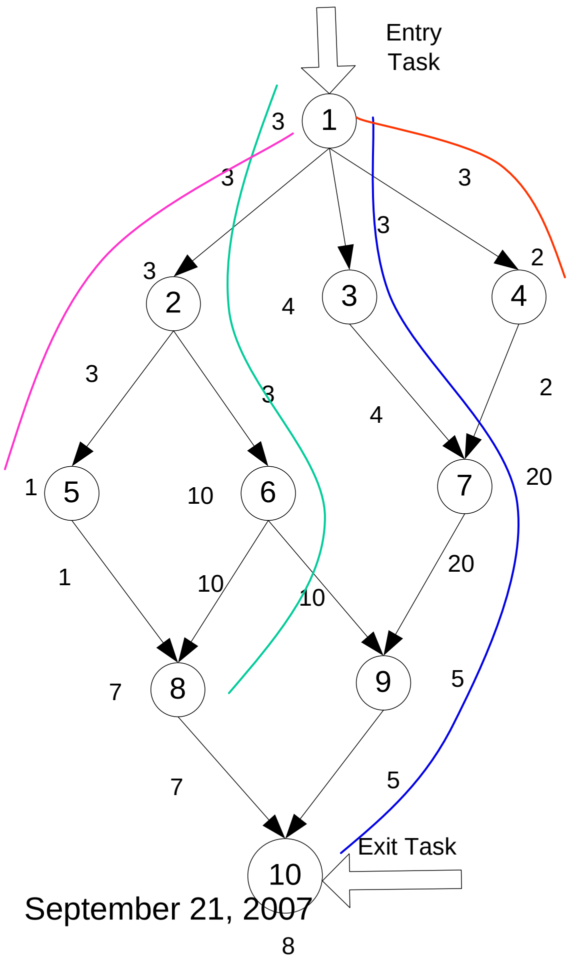
FP: Favorite Predecessor

S. Darbha and D. P. Agrawal, "A Task Duplication Based Scalable Scheduling Algorithm for Distributed Memory Systems", ***J. Parallel and Distr. Comp., vol. 46, no. 1, pp. 15-27, Oct. 1997.***

Energy-Efficient Scheduling

- Implementation (An Example) Cont.

Step 3: Generate original scheduling List & find all critical paths



Task	Level	EST	ECT	LAST	LACT	FP
1	40	0	3	0	3	--
2	28	3	6	4	7	1
3	37	3	7	3	7	1
4	35	3	5	3	5	1
5	16	6	7	16	17	2
6	25	6	16	7	17	2
7	33	7	27	7	27	3
8	15	16	23	18	25	6
9	13	27	32	27	32	7
10	8	32	40	32	40	9

Original Scheduling List: {10, 9, 8, 5, 6, 2, 7, 4, 3, 1}

Energy-Efficient Scheduling

- Implementation (An Example) Cont.

Step 4: Task duplication decision and allocation

In the first iteration, scheduling List is {10, 9, 8, 5, 6, 2, 7, 4, 3, 1} T10 is the starting task.

To save time, EETDS will always try to allocate tasks in the same critical path to the same processor, which means…

Allocate T10 → T9 → T7 → T3 → T1 to processor1

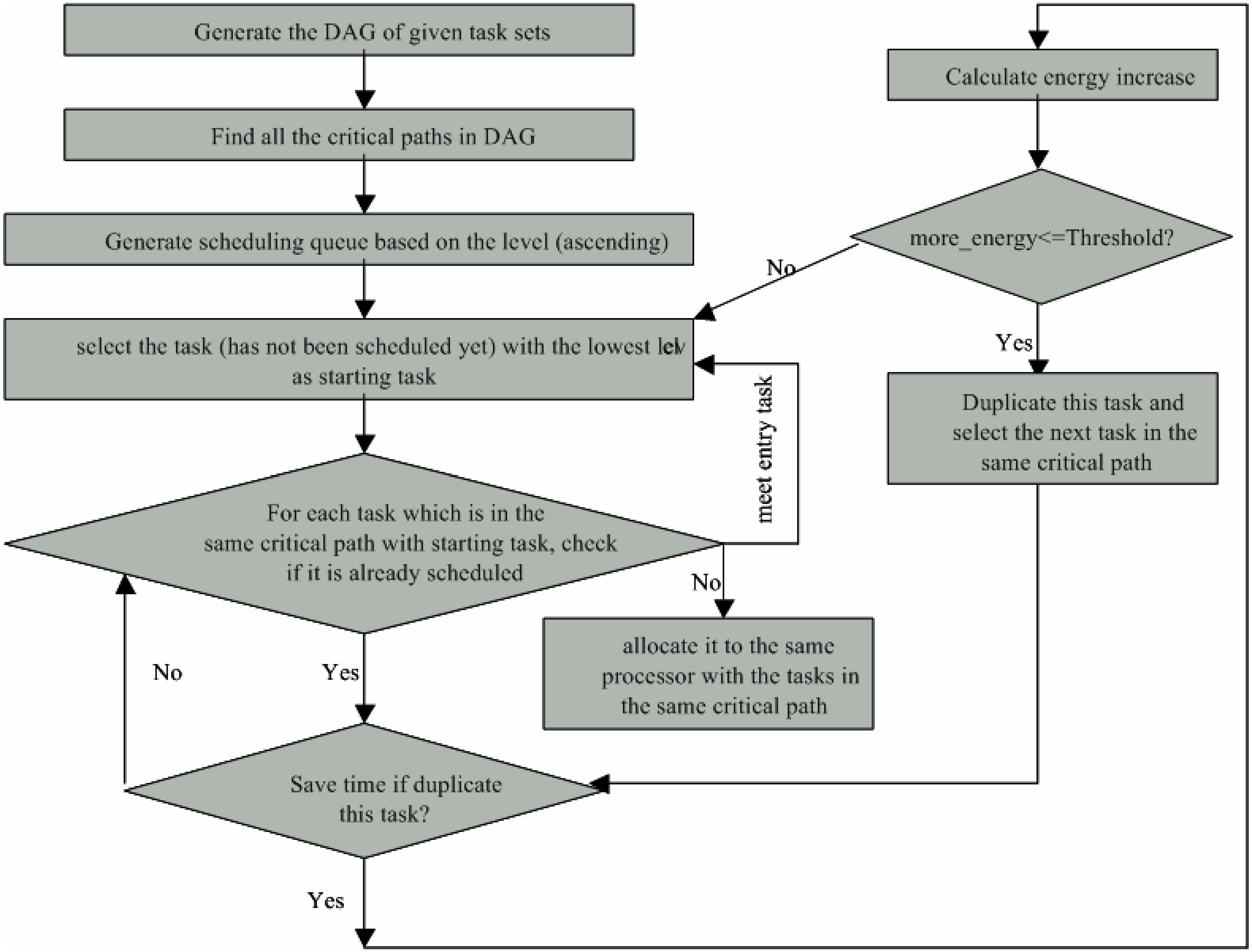
Now EETDS meets entry task T1, it will enter the second iteration

scheduling List is {10, 9, 8, 5, 6, 2, 7, 4, 3, 1} then T8 should be the starting task, it will try to Allocate T8 → T6 → T2 → T1 to processor2

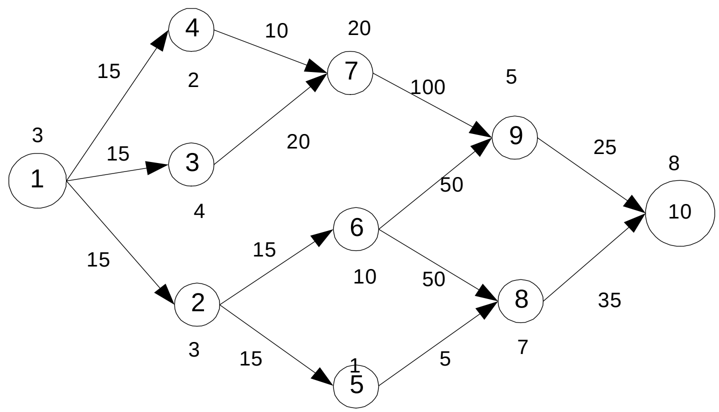
However, T1 has already been allocated to processor1, duplicate or not?

Time factor: **LAST(v2) - LACT(v1) = 4 – 3 = 1 < CC12 = 3** , so we can save 2s if duplicate T1 Energy factor: **energy_increase = 44w×3 – 33.6w×3 = 31.2J > Threshold =25J**

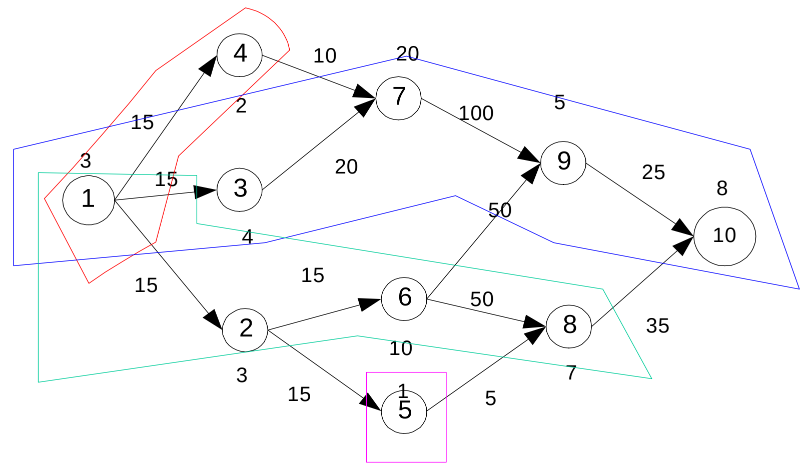
No duplication for T1 because of too much energy consumption, even we can save time


Generate the DAG of given task sets
Find all the critical paths in DAG
Generate scheduling queue based on the level (ascending)
select the task (has not been scheduled yet) with the lowest lev
as starting task
For each task which is in the
same critical path with starting task, check
if it is already scheduled
meet entry task
No
allocate it to the same
processor with the tasks in
the same critical path
No
Yes
Save time if duplicate
this task?
Yes
Calculate energy increase
more_energy<=Threshold?
No
Yes
Duplicate this task and
select the next task in the
same critical path

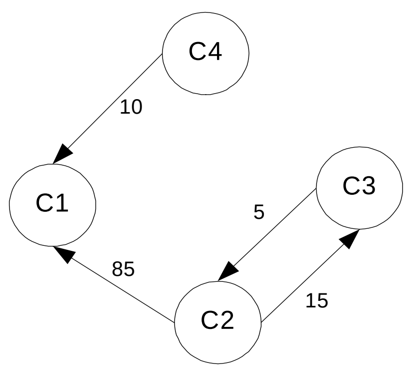
Energy-Efficient Scheduling



(a) The originial task description



(b) The partitioned task graph



(c) The cluster graph

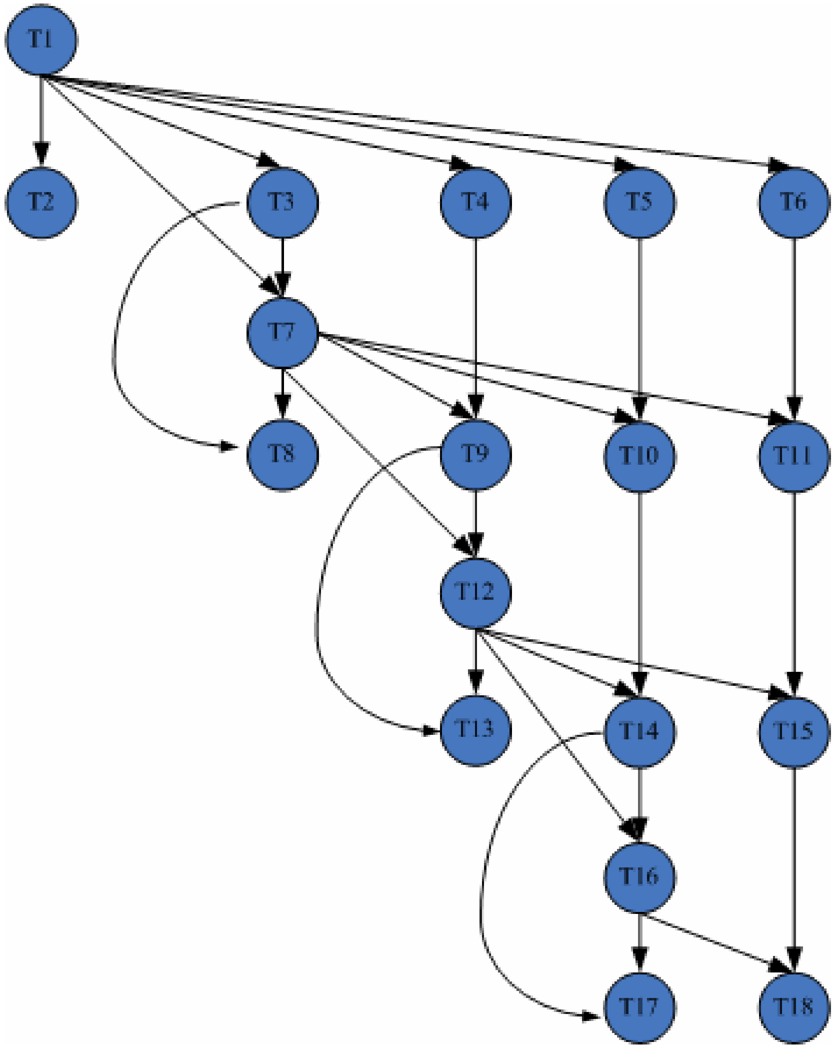
Cluster 1 is allocated to node C

Cluster 2 is allocated to node B

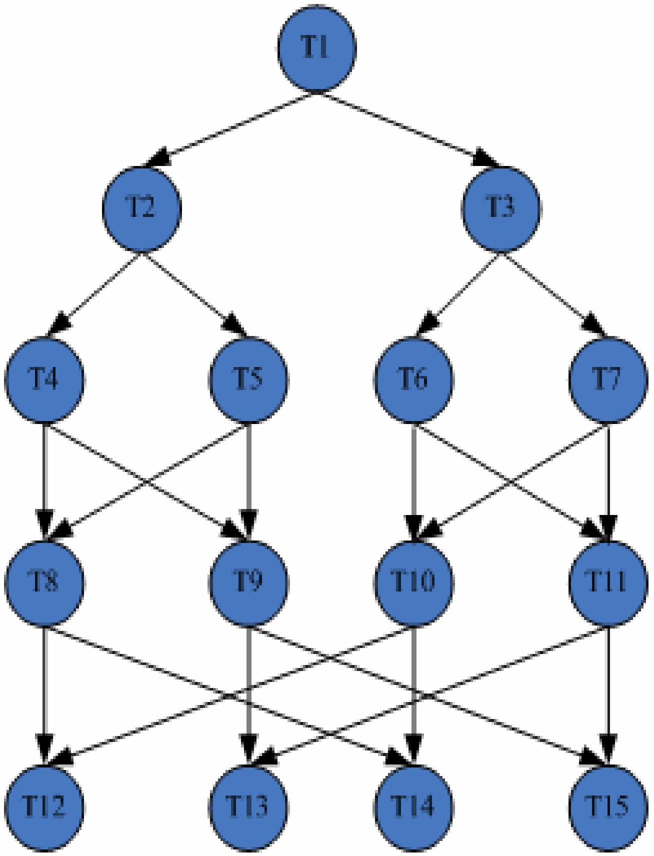
Cluster 3 is allocated to node D

Cluster 4 is allocated to node A

(d) Final allocation list



Gaussian Elimination Task

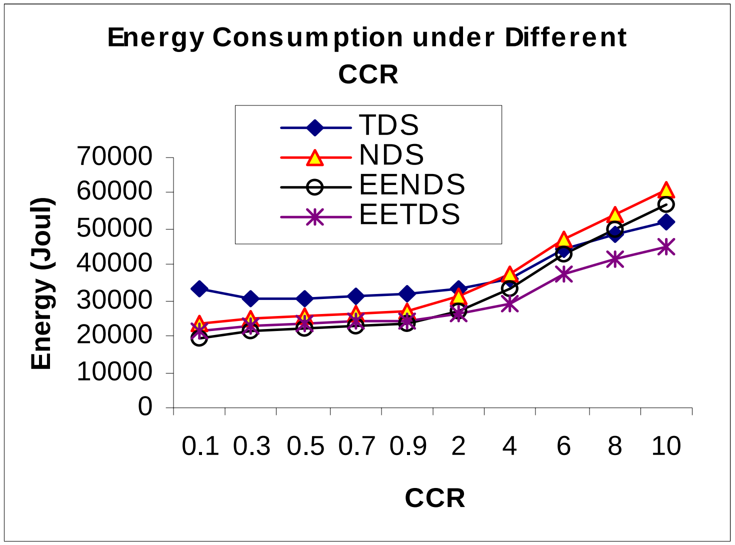


Fast Fourier Transform Task

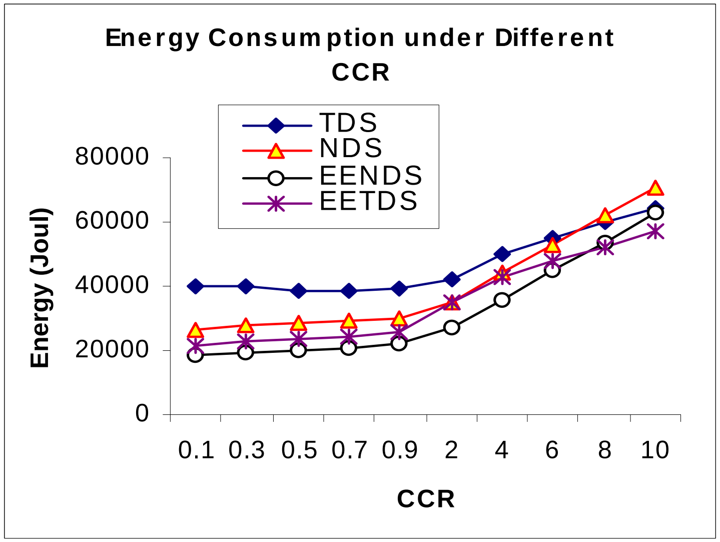
Simulation Results

- Impact of CCR

CCR = Average Communication Time / Average Computation Time



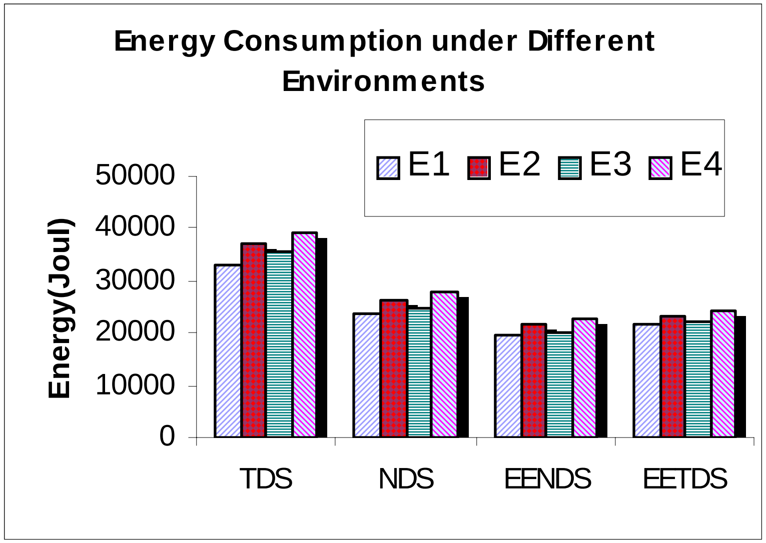
CCR Sensitivity of Gaussian Elimination



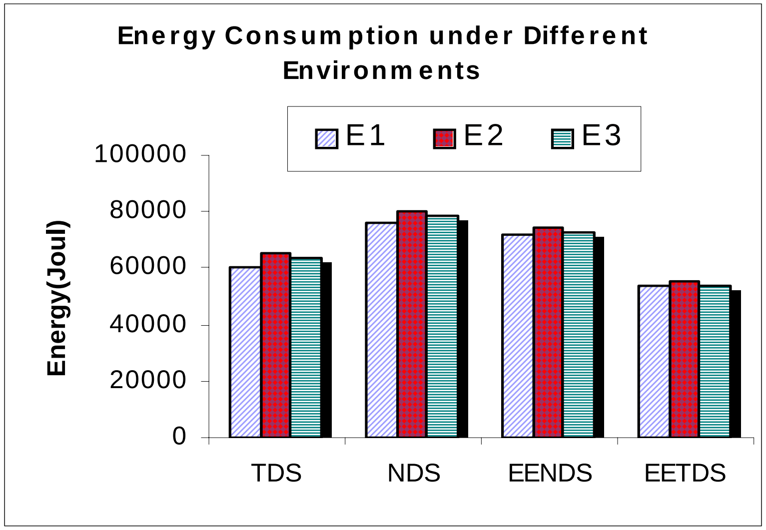
CCR Sensitivity of FFT Elimination

Simulation Results

- Impact of processor types



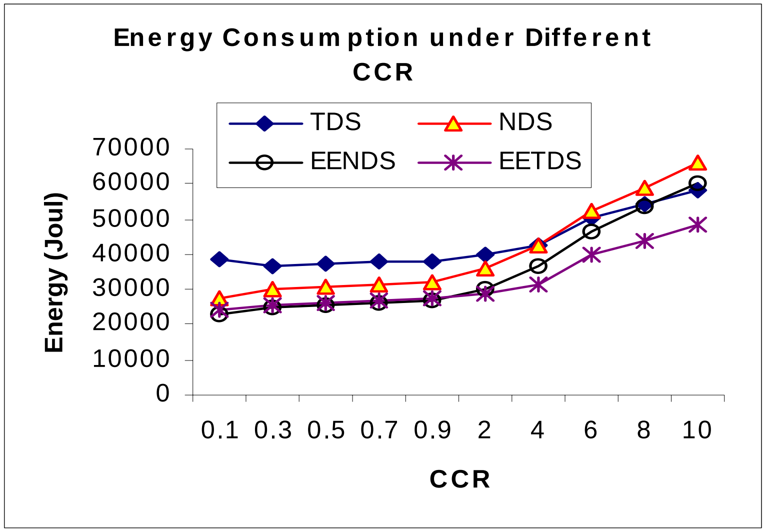
Energy consumption for Gaussian when Net_Energy=60 and CCR=0.1



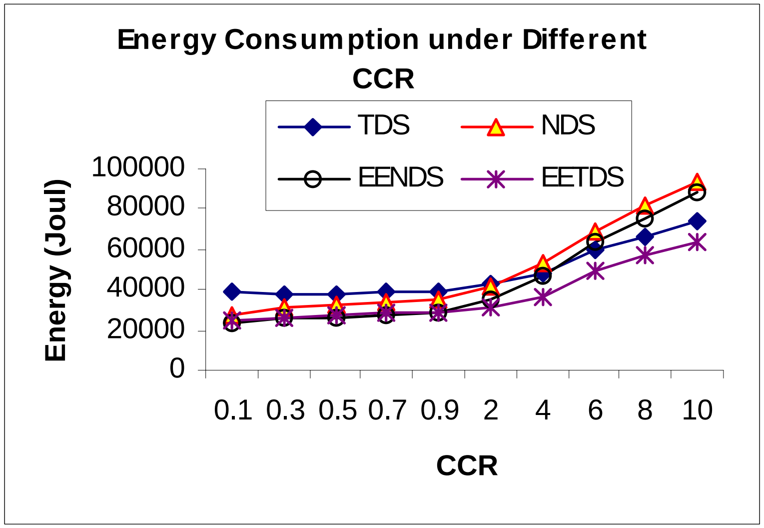
Energy consumption for Gaussian when Net_Energy=60 and CCR=8

Simulation Results

- Impact of networks



Energy consumption of Gaussian (Net = 33.6W)



Energy consumption of Gaussian (Net = 60W)

Thank you!

Questions?