SST Competition Instructions

Problem description

The research department at your company has developed a new application, but it runs terribly on their outdated hardware, so they've been given $3500 to purchase a new machine. As the chief computer architect, they've asked you to advise them on what to buy. You have received some quotes for different systems, but there are many options to pick from and each has a different cost (see table below). The company wants the **most cost efficient system (highest performance per cost)**. You need to use SST to determine which configuration, within the budget, meets the goal of the company. To help you, the research department has modeled their workload as a Miranda workload generator subcomponent.

Instructions

1. Install SST 9.1 from **www.sst-simulator.org**. You will need sst-core and sst-elements.
2. Edit the Makefile in the **workload/** directory so that *SSTELEMSOURCE* and *SSTCOREINSTALL* point to your sst-elements source root directory and sst-core install root directory, respectively. Then run `make`. You can run `sst-info sc19` to check that SST can now find the new subcomponent.
3. The SST input script **workload/scc-sst-node.py** is parameterized to take each of the system options in the table. Comments at the top of the script describes the simulated architecture in some more detail. Do not modify the simulated architecture or workload in the script (specifying partitioning or printing additional information is OK). The script can be run as follows:

```
sst scc-sst-node.py --model-options='-n=X0 -c=X1 -t=X2 -l1=X3 -l2=X4 -s=X5 -l3=X6 -b=X7 -w=X8 -m=X9'
```

   where the `Xs` should be replaced with an option from the table. For example:

```
sst scc-sst-node.py --model-options='-n=40 -c=fast -t=no -l1=big -l2=small -s=private -l3=small -w=6 -b=slow -m=basic'
```

4. The script will immediately reject and not run any configuration that exceeds the $3500 limit with the message "ABORT: Cost exceeds limit of 3500. Cost is ...". If the --model-options string you give is valid and within the budget, the script will print the configuration and cost, and the simulation will run.
5. A text file has been provided in this directory for you to record your answers. Fill in the configuration (--model-options string), simulated time, and cost for the highest performance/cost system (cost.txt). Correct answer (this answer is unique) receive all 35 points; incorrect answers receive up to 30 points prorated by how closely they come to the correct system. Invalid configurations, configurations that exceed the budget, and answers with an incorrect cost or simulation time will receive no points. For simulation time, include all digits and units reported by SST (e.g., "32.3187 us").

> **Scoring formula for non-correct answer:**
>
> $$\frac{simulated\ time_{best} * cost_{best}}{simulated\ time_{submitted} * cost_{submitted}} * 30$$

Table

Blue indicates arguments the script accepts. For example, the options for core type are -c=slow, -c=medium, or -c= fast.

Category	Script param					
Core count	-n	22	24	30	36	40
Core type	-c	slow	medium	fast		
		$15 / core	$23 / core	$40 / core		
		1.8GHz 16 outstanding memory requests	2.5GHz 16 outstanding memory requests	4GHz 32 outstanding memory requests		
SMT	-t	no	yes			
		free	1.7 * core cost			
		1 hardware thread/core	2 hardware threads/core			
L1 size (per core)	-l1	small	big			
		$18 / core	$28 / core			
		16KB, 8-way, 1 cycle access	64KB, 16-way, 4 cycle access			
L2 size (per core)	-l2	small	big			
		$14 / core	$20 / core			
		128KB, 8-way, 5 cycle access	512KB, 16-way, 7 cycle access			
L2 organization	-s	private	shared			
		free	$2 / core			
		private	shared			
L3 size (per core)	-l3	small	big			
		$20 / core	$24 / core			
		1MB, 16-way, 10 cycle access	1.5MB, 32-way, 14 cycle access			
Network-on-chip	-b	slow	fast			
		$5 / core	$9 / core			
		1.6GHz	2.2GHz			
Memory channels	-w	6	8			
Memory type	-m	basic	bw	lat		
		$110 / channel	$200 / channel	$260 / channel		
		Baseline	2X bandwidth	~1/3 lower latency		