Mega-Scale Computing

Based on Low-Power Technology and Workload Modeling

MEGASCALE

http://www.para.tutics.tut.ac.jp/megascale/

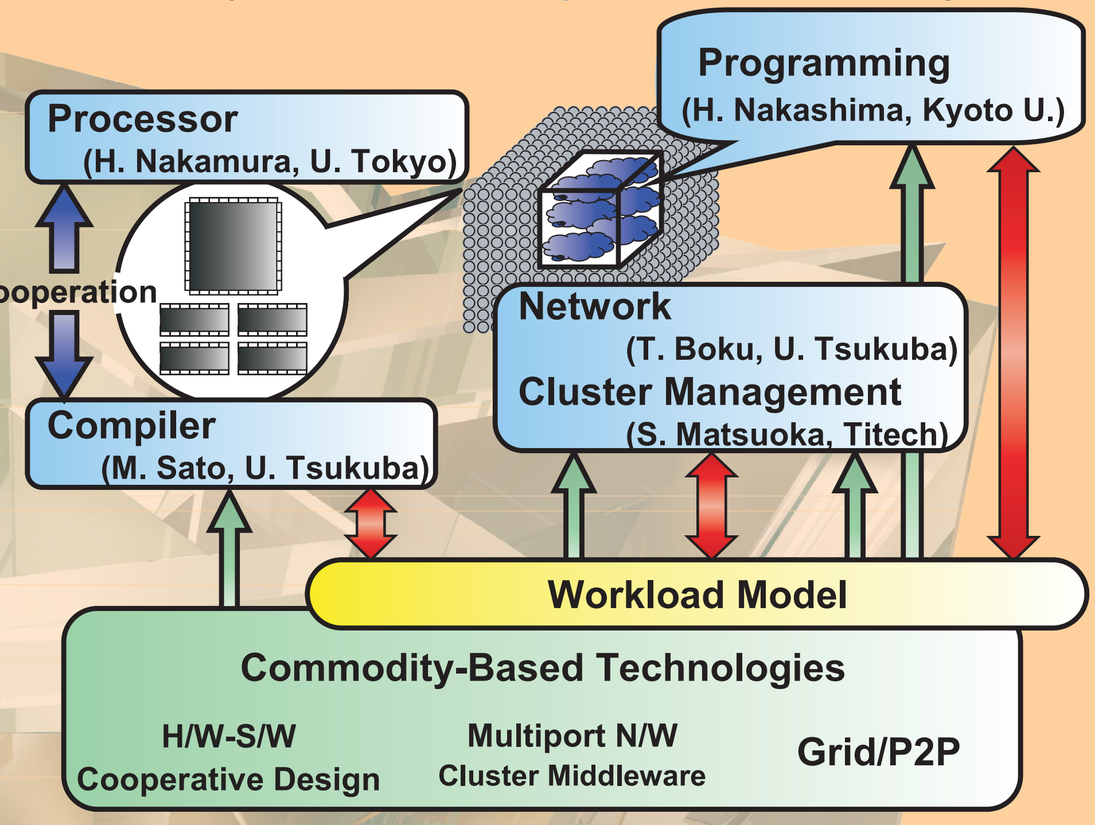
Introduction

When we started our *mega-scale computing* project five years ago, the HPC world was led by high-performance but *hot* processors and HPC-dedicated technology. At that time we dared to claim that Peta-FLOPS should be achieved only by million-scale aggregation of *low-power* processors built with *commodity-based* technologies and with deep awareness of *dependability* and *programmability*. After five years, it is very sure now that our claim is proved correct with the facts that leading HPC systems today and near-future have a considerably large number of relatively small and power-aware cores. This means our research outputs, from processor architecture to programming environments, now have the field and era to flourish in.

Research Outcomes

In our research, supported by a JST/CREST program entitled “Advanced Information Technology for Future C&C Society and Community”, we proposed, designed and implemented various novel technologies for *mega-scale computing*, and integrated them on our 10^3-scale prototype named *MegaProto*. The major outcomes from five research groups are as follows.

- Processor group designed a low-power/high-performance architecture *SCIMA,* its power-aware compiler, and *adaptive power control* for low-power clusters.
- Compiler group gave two power-aware techniques; *profile-based compilation* to minimize PD or ED; and *slack reclamation* for parallel tasks represented by DAGs.
- Network group developed two commodity-based network technologies; *RI2N* with multiple links for high performance/reliability; and *VFREC-Net* to exploit multiple paths for large bandwidth by VLAN technology.
- Cluster management group built a FT-MPI framework *Cuckoo* with which various FT techniques such as *model-based fault detection* will be integrated.
- Programming technology group proposed a performance modeling technique given by a meta-program written in a parallel script programming language *MegaScript*



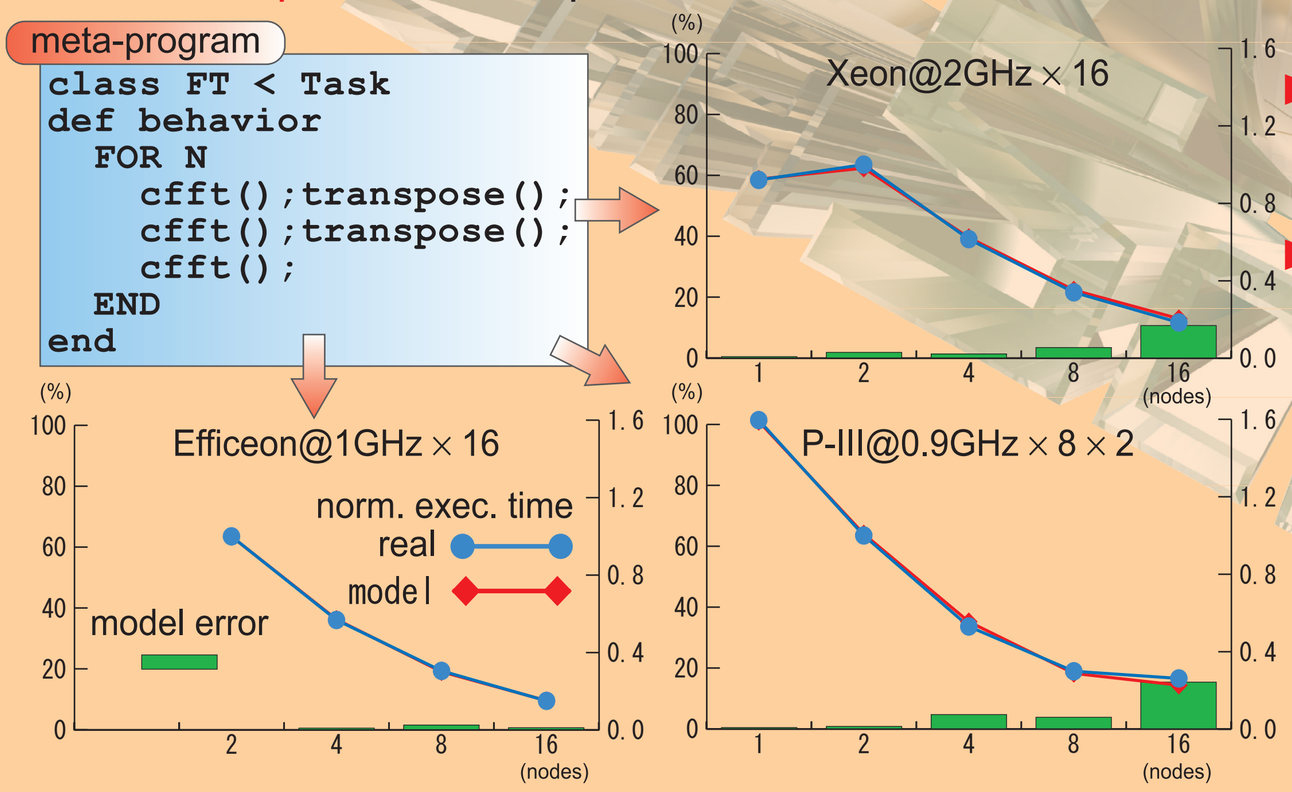
* *Highlighted* research topics are presented in posters.

MegaScript Language

- Two-tier task-parallel programming.
- Easy-to-write script language.
- Meta-programming for *workload modeling.*
- *Model adaptation* to computation environments.

meta-program

```
class FT < Task
def behavior
  FOR N
    cfft();transpose();
    cfft();transpose();
    cfft();
  END
end
```



Model-Based Fault Analysis

- *Trace calls/returns* and their timing by dynamic instrumentation on each node.
- Map traces of nodes onto *vector space* each dimension of which corresponds to execution time fraction of each function.
- Calculate *suspect score* of each vector as the distance from its *k*-th (e.g. 2nd) nearest neighbor to detect high score node as *anomaly*.
- A rarely-occurred malfunction in SCore running on Titech 129-node cluster was *detected* with its cause.