PaaS for creating and executing concurrent cloud applications
OVERVIEW

1) Fundamental transformation in computing:
   ▪ Concurrent apps on dynamically shared resources
     ▪ Micro-services: *unpredictable* bursts of short-lived transactions

2) Industry-wide, must-solve challenge:
   ▪ Calls for an open-source platform approach

3) Enabling performance-critical cloud computing:
   ▪ Dynamic parallel execution environment (DPEE)

4) Call for collaboration:
   ▪ Open-source development environment for creating concurrent apps for effective cloud execution
1) Fundamental Transformation in Computing

A. What higher clock rates were needed for (=everything), concurrent processing will be needed for:

- For decades, application program speed-up was automatic through higher processor clock rates
  - Processor clock rate increases no longer feasible/economical
- Going forward performance gain demands concurrent execution
  - Including at intra-application level
    - Very complex and distracting!
1) Fundamental Transformation in Computing

B. Computing is increasingly taking place on dynamically shared cloud infrastructure

- A. (concurrency) and B. (cloud computing) together:
  → Need for dynamic parallel computing

- Must-solve challenge for application developers, who however are ill-equipped to address it

→ Fundamental challenge for software industry calling for a solution
1) Concurrent Apps on Dynamically Shared Resources

Need to accommodate:
- Apps with variable inter-task topologies
- Dynamic # of active instances per app
- Dynamic # of active tasks per app instance
- Inter-task communication among dynamically scheduled and placed tasks

App instances/tasks increasingly short-lived microservices/actors → Need for very rapid dynamic resource management → Would increase system software overhead ↔ No longer tolerable as hardware is not getting faster!!!
1) Overhead of Optimal Resource Management

- Optimizing usage of a pool of resources (e.g. 1000s of cores) among:
  - multiple (10s) of apps, with their respective resource entitlements,
  - dynamic number (10s) of active instances per app, and
  - dynamic selections of (10s) active tasks per app instance ..

- where the dynamic sets of active instances and tasks vary at network packet / inter-task communications (ITC) message transmission time granularity (i.e. at ~ 1kb/1Gpbs = microsecond granularity) ..

- requires identifying for each of the 1000s of cores the optimal app instance task among the 1000s of alternatives 1M times per second
  - and carry out the actual resource access re-configurations etc. ..

- i.e. performing Billions of complex computations + re-conf:s / second

If done in software, optimal resource management itself would demand most of the resources, defeating its purpose!
  - Static allocation neither a viable approach, given the increasingly dynamic & unpredictable load variations for enterprise/web cloud apps
1) Scalability **Problem** in Parallel Cloud Computing = **SYSTEM SOFTWARE OVERHEAD**

Scaling parallel cloud computing with **software operating system**

Concurrency overhead increases according to:

\[ f_1(\text{#apps}) \times f_2(\text{#tasks/app}) \times f_3(\text{#cores}) \]

**Scaling factor:** 
\[ \text{# apps} = \text{# tasks/app} = \text{# cores} \]
1) Scalability **Solution** for Parallel Cloud Computing = AUTOMATE SYSTEM FUNCTIONS IN HARDWARE

Scaling parallel cloud computing with **hardware operating system**

Eliminating parallelization software overhead by hardware OS allows near-linear application on-time throughput scaling

**Throughput, overhead / normalized uncore throughput**

- **System throughput capacity**
- **Overhead rate**
- **Throughput rate/core**

**Scaling factor:** # apps = # tasks/app = # cores
2) Industry-wide, Must-solve Challenge

- Existing computing tech suppliers offer point-products or half-solutions:
  - parallel programming languages (extensions), frameworks, tools, middleware, manycore processors, etc.

- Burden of cost-efficient concurrent cloud computing left for individual application developers (SaaS vendors)
  - .. or worse yet, to the app users (SaaS customers)!

→ Status-quo not sustainable

→ A holistic platform solution needed for parallel cloud computing
3) Parallel Cloud Computing Platform

- Existing parallel computing tools mainly limited to parallel *programming* aspect of parallel computing challenge
- Any parallel *execution* technologies etc. designed for acceleration on dedicated machines, not on the cloud
- Parallel execution in cloud i.e. *dynamic parallel execution*, though critical for cloud computing cost-efficiency, left unaddressed by legacy vendors

→ Dynamic parallel execution *critical* piece of performance-critical cloud computing
ThroughPuter introduces a critical technology enabling effective parallel cloud computing platform:

- Dynamic parallel execution environment (DPEE)
  - Implemented in HDL code
- User-friendly PaaS business model
  - Incentives to maximize computing on-time throughput performance and resource efficiency as well as user’s productivity
- Intellectual property rights for dynamic parallel execution
  - 44 patents issued and pending worldwide, incl. 26 granted US/UK patents
3) ThroughPuter: Platform Overview

User interface and development tools:
• Represent the client apps as concurrently executable tasks

Parallelized application tasks (executables)

Application tasks, residing in their dedicated memory segments

• From each application: core demand expressions, task priority lists
• For each app: sets of tasks for execution

Hardware operating system and dynamic on-chip network:
• Switch app tasks to cores based on processing loads and contractual policies
  • Dynamically, connect tasks of any given app, rather the cores statically
  → Simpler implementation, higher performance

© ThroughPuter, Inc. All rights reserved.
3) ThroughPuter: Open Parallel Computing PaaS

Reasons for open-source collaboration for parallel computing PaaS to leverage ThroughPuter’s dynamic parallel execution environment (DPEE):

1) **Less low-level work**: DPEE automates dynamic resource management etc. parallel execution OS functions in (programmable) hardware, providing higher level interface (API) for the development environment software.

2) **Higher performance** due to minimum-overhead hardware automation of system tasks such as optimally allocating processing capacity, scheduling and placing application tasks for execution, inter-task communications, billing etc.

3) **Built-in cloud computing security**: mechanisms for unauthorized interactions between different applications simply non-existent in the hardware.

4) **Open standard interface** between development and execution environment.

*Dynamic parallel capacity allocation example and the enabling cloud processor architecture block diagram on next slides ➔*

Open parallel computing PaaS needed -- ThroughPuter execution environment goes a long way toward reaching that goal.
3) Dynamic Core Allocation Example

Core Allocation Period (CAP)

All tasks continuing on consecutive CAPs stay on their existing core, and continue processing uninterruptedly through CAP boundaries.

Any application can burst even up to full capacity of the shared core pool, as long as actually materialized core demands by other apps are met up to their entitlements.

Any task can communicate with all other tasks of its app instance without knowledge of whether or at which core any given task is running.

Allocation of cores re-optimized among applications for each new CAP based on core demands and entitlements of the applications.
3) Concurrent cloud computing challenge - Technical

Example:
Average core demands by applications sharing a 16-core processor

- app1: 12.5%
- app2: 25%
- app3: 12.5%
- app4: 50%

Actual core demands by applications over 10us period

Actual and average core demands at t = 6 microseconds

Cores vs. Application #

Idle capacity

Blocked demand
• The actual, momentary processing capacity demand by any given individual application program hardly ever equals its ‘average’ demand

→ Non-adaptive capacity partitioning leads to wasting of resources and blocking on-time throughput

• Capacity being held statically in reserve for idling applications should have been allocated to other applications on the manycore processor that at that time would have been able to use it
3) Concurrent cloud computing challenge - Solution

Dynamic Parallel Execution Environment (DPEE) of ThroughPuter PaaS enables realtime application load adaptive, resource efficient and high performance cloud computing.
3) DPEE: App load adaptive manycore architecture

**STEP 1**
Once per a core allocation period (e.g., microsecond):
Allocate cores to applications

**STEP 2**
For each application:
Select to-be-executing tasks
Ready-task priority ordered lists from applications, along with the core types demanded by each task

**STEP 3**
For each application:
Map selected tasks to available cores

**Billing subsystem**
For each application:
Core entitlements
For each application:
Billables

**Fabric network and memories**

- core slot
- core slot
- core slot
- core slot
3) DPEE: Multi-stage manycore architecture

- Real hardware resources for ITC, memory/network I/O, load balancing → No software abstractions (=overhead) needed
- Hardware automated support for any mixes/matches of data flow topologies among apps sharing the multi-stage manycore cloud processors
- Further scalability via load balanced parallel arrays of the cloud processors
3) Example application data flow topologies

- Scatter-gather
- Pipelined
- Hybrids of above
- Any parallel mixes/matches of above

The hardware automatically adapts to (potentially dynamic) data flow topology of any given application, without any system software (re)configurations
3) Summary of Advantages

- **PERFORMANCE and COST-EFFICIENCY:**
  - Architecturally maximized application processing on-time throughput per unit cost
    - Hardware operating system and on-chip network optimized for execution of multiple concurrent apps on dynamically shared manycore processors

- **SECURITY:**
  - Full isolation, right from hardware level up, among client applications dynamically sharing a pool of cores

- **PRODUCTIVITY:**
  - PaaS automates concurrent cloud app development + deployment

- **OPEN SYSTEM:**
  - PaaS software and processor-core hardware to be open-sourced
  - Host anywhere; ThroughPuter commercial hosting an option
4) Call for Collaboration

• The need for parallel processing an emerging, MAJOR industry and profession wide challenge
  → Open-source collaboration a natural approach

• Need for architectural optimization across traditional application, system and hardware layer boundaries
  → SOLUTION: Open-source PaaS reaching all the way to parallel cloud computing optimized hardware

– ThroughPuter’s contribution: Hardware architecture designed for dynamically shared multi-tenant parallel cloud computing
  • Secure hardware OS for manycore fabric with on-chip network, taking care of dynamic capacity allocation, parallel program execution mgmt

– Collaboration opportunities:
  • Development environment and tools
  • Extensions of the PaaS for specific user domains: channel partnerships
  • Processor-core IP for the cloud processors
  • Physical hardware supply, physical hosting (IaaS) etc.