The Design of a CCA Framework with Distribution, Parallelism, and Recursive Composition

Francisco Heron de Carvalho Junior
Ricardo Cordeiro Correa

Research Group On Parallelism, Graphs, and Optimization

Mestrado e Doutorado em Ciência da Computação (MDCC)
Universidade Federal do Ceará
Fortaleza, Brazil
Topics

- Parallelism in component models for HPC;
- The Hash Component Model;
- HPE – Hash Programming Environment;

Why to make HPE a CCA framework?

How to make HPE a CCA framework?

What HPE brings to the world of CCA frameworks?

Conclusion / Further Works.
Topics

• Parallelism in component models for HPC;
  • The Hash Component Model;
  • HPE – Hash Programming Environment;
  • Why to make HPE a CCA framework ?
  • How to make HPE a CCA framework ?
  • What HPE brings to the world of CCA frameworks ?
  • Conclusion / Further Works;
Component Models in HPC

• CCA (Common Component Architecture)
  – Frameworks may investigate forms of parallelizing components;
  – CCAffeine (parallel), XCAT (distributed), DCA (distributed/parallel), SciRun2 ($M \times N$ couplings);
  – SCMD, cohorts, MCMD, teams, PRMI ...

• Fractal/GCM (Grid Component Model)
  – Distributed by design (grid computing);
  – Collective interfaces ($multicast$, $gathercast$, $M:N$) for parallelism;
  – $M \times N$ couplings – alternatives for implementation:
    • Composition of $gathercast$ and $multicast$;
    • Controllers;
Topics

• Parallelism in component models for HPC;

• **The Hash Component Model**;

• HPE – Hash Programming Environment;

• *Why* to make HPE a CCA framework?

• *How* to make HPE a CCA framework?

• *What* HPE brings to the world of CCA frameworks?

• Conclusion / Further Works.

MDCC – Mestrado e Doutorado em Ciência da Computação
The Hash (#) Component Model

- A (meta-)model of parallel components;
- A #-component is formed by a set of units;
- Each unit in a computing node of distributed platform;
- Hierarchical composition by overlapping;
- Concern-oriented parallel programming;
  - Concerns viewed as orthogonal to processes;
- Kinds of #-components
  - Different component/connection/deployment models;
  - # programming systems.
The Hash (#) Component Model – Overlapping

MDCC – Mestrado e Doutorado em Ciência da Computação
Topics

• Parallelism in component models for HPC;
• The Hash Component Model;
• **HPE – Hash Programming Environment**;
• **Why** to make HPE a CCA framework?
• **How** to make HPE a CCA framework?
• **What** HPE brings to the world of CCA frameworks?
• Conclusion / Further Works.
The Hash Programming Environment – HPE

• A general purpose # programming system for cluster computing;
• Kinds: architectures, environment, data structures, computations, synchronizers, applications, qualifiers;
• Hash Type System (HTS):
  – Abstract components (contracts, or types, for #-components);
  – Supports context-based dynamic binding;
• Architecture:
  – Front-End: life-cycle control;
  – Back-End: execution platforms;
  – Core: repository;
• Hosted at Google code (http://hash-programming-environment.googlecode.com).
The Hash Programming Environment – HPE

Farm

\[\text{input type} = I : \text{Data},\]
\[\text{scatter strategy} = S : \text{Distribute}[\text{source} = I, \text{target} = J],\]
\[\text{job type} = J : \text{Data},\]
\[\text{work} = W : \text{Work}[\text{input} = J, \text{output} = R],\]
\[\text{result type} = R : \text{Data}\]
\[\text{gather strategy} = G : \text{Collect}[\text{source} = R, \text{target} = O],\]
\[\text{output type} = O : \text{Data}\]

Component-Oriented Skeletal Programming

CONTEXT

MDCC – Mestrado e Doutorado em Ciência da Computação
The Hash Programming Environment – HPE

MDCC – Mestrado e Doutorado em Ciência da Computação
The Hash Programming Environment – HPE

```
Farm [ input type = Data, output type = Data,
      job type = Data, result type = Data,
      scatter strategy = Distribute[source = Data, target = Data],
      work = Work[input = Data, output = Data],
      gather strategy = Collect[source = Data, target = Data]]
```

**VeryGenericFarm**

who it implements

scatter

work

gather

who it needs

input

job

result

output

MDCC – Mestrado e Doutorado em Ciência da Computação
The Hash Programming Environment – HPE

**NumericalIntegrator**

- **method** = Romberg,
- **function** = MyFunction

who it implements

**Farm**

- **input type** = IntegralCase[MyFunction],
- **output type** = Double,
- **job type** = List[IntegralCase[MyFunction]],
- **result type** = List[Double],
- **scatter strategy** = DistributeInterval[source = IntegralCase[MyFunction], target = List[IntegralCase[MyFunction]]],
- **work** = Work[input = List[IntegralCase[MyFunction]], output = List[Double]],
- **gather strategy** = Collect[source = List[Double], target = Double] who it needs

**MyRombergIntegrator**

MDCC – Mestrado e Doutorado em Ciência da Computação
The Hash Programming Environment – HPE

- HPE x C#/MPI.NET
- Multi-Dimensional Numerical Integration using a Farm;
- Linux-Based Cluster
  - [http://castanhao.lia.ufc.br](http://castanhao.lia.ufc.br)
  - 28 nós (12 operational);
- Binding time negligible;
  - Constant on number of processors and problem size;

---

MDCC – Mestrado e Doutorado em Ciência da Computação
Topics

- Parallelism in component models for HPC;
- The Hash Component Model;
- HPE – Hash Programming Environment;
- **Why to make HPE a CCA framework?**
- **How to make HPE a CCA framework?**
- **What HPE brings to the world of CCA frameworks?**
- Conclusion / Further Works.
Why to Make HPE a CCA Framework

• To give evidence about the expressiveness of the hash component model;

• Communicate novel ideas;

• To contribute to the design of CCA frameworks;
  – A general notion of parallel component (MCMD by design);
  – A more expressive way to integrate parallelism and distribution;
  – More control over performance of connections (always direct);
  – A new perspective of hierachical and recursive composition;

• To make case studies with existing CCA applications.
Topics

• Parallelism in component models for HPC;
• The Hash Component Model;
• HPE – Hash Programming Environment;
• Why to make HPE a CCA framework?
• How to make HPE a CCA framework?
• What HPE brings to the world of CCA frameworks?
• Conclusion / Further Works.
How to Make HPE a CCA Framework – Ports

provides

#-component C

takes

impl

uses_1

uses_2

…

uses_k

inner components
How to Make HPE a CCA Framework – Ports

provides_1
provides_2
provides_n

uses_1
uses_2
uses_k

C

implement

inner components

provides_1
provides_2
provides_n
provides ports

uses ports

CCA look of
#-component C

provides ports

uses ports

MDCC – Mestrado e Doutorado em Ciência da Computação
How to Make HPE a CCA Framework – Binding

processing node #1

processing node #2

processing node #k

direct binding

message passing

MDCC – Mestrado e Doutorado em Ciência da Computação
How to Make HPE a CCA Framework – **Binding**

**binding is an inner component of the client**

**C**

- **u_1**
- **u_2**
- **u_m**

**binding**

- **c_1**
- **c_2**
- **c_m**

**server**

- **v_1**
- **v_2**
- **v_n**

**S**

**binding is an inner component of the server**

**rendezvous indirect binding (MxN coupling)**

**binding is of kind SYNCHRONIZER**

**direct binding**

- **message passing**

- **c_1**

- **c_2**

- **c_m**

**client node #1**

- **u_1**

**client node #2**

- **u_2**

**client node #m**

- **u_m**

**server node #1**

- **s_1**

**server node #2**

- **s_2**

**server node #n**

- **s_n**

**B**

**binding is an inner component of the client**

**binding is an inner component of the server**

**binding is of kind SYNCHRONIZER**

**MDCC – Mestrado e Doutorado em Ciência da Computação**
How to Make HPE a CCA Framework – Binding

(a) involved #-components

- Binding is of kind SERVICE

(b) run-time connections

MDCC – Mestrado e Doutorado em Ciência da Computação
How to Make HPE a CCA Framework – Instance

(a) class of $s_i$

override void createSlices {
... 
DGAC.createSlice( ... $p_i$ ... );
... 
DGAC.createSlice( ... $q_i$ ... );
... 
}
...
How to Make HPE a CCA Framework

CCA PARALLEL FRAMEWORK

Manager

Worker #1
Worker #2
Worker #n

Unit-Level BuilderService's Parallel Component View

message passing channel for communication between units of a CCA component

Back-End Web Service

DGAC

Component-Level BuilderService

resolution
loading configurations

deployment

ComponentRepository

DDAO (DGAC Database Access Object)

DGAC database access interface

DGAC Database

resolution
loading configurations

deployment

DGAC database access interface

Single Unit View

Parallel Component View

...
Topics

- Parallelism in component models for HPC;
- The Hash Component Model;
- HPE – Hash Programming Environment;
- **Why** to make HPE a CCA framework?
- **How** to make HPE a CCA framework?
- **What** HPE brings to the world of CCA frameworks?
- Conclusion / Further Works.
The features of the CCA framework HPE

Generalized Distributed Parallel Components

- **MCMD** (Multiple Component Multiple Data);
- A parallel component addresses a single concern;
  - (CCAffine) A cohort of components is a single #-component;
- Communication is encapsulated in #-components;
  - No clandestine communication (synchronizer #-components encapsulating MPI);
  - Components are decoupled with respect to parallel interaction;
- Expressiveness:
  - No restrictions for composition of parallel components;
  - Comparable to MPI.
  - Parallel programming with skeletons;
  - Connectors as #-components.
The features of the CCA framework HPE

**M × N Couplings**

- Binding \#-components, encapsulating “logic of transfers”;
  - Bindings can be tuned according to the architecture features;
  - This is not true for PRMI without imposing restrictions on distribution semantics and on the allowed data types;
  - The same for GCM collective interfaces;

- The client and the server may be decoupled!
  - A server may attend distinct clients through distinct bindings;
  - A client may connect to distinct servers through distinct bindings;

- The client and the server may share a communicator.
The features of the CCA framework HPE

Recursive Composition

• Dealing with complexity of large scale component ensembles;
  – A form to achieve abstraction and modularity;
• Vertical (*is-inner-of*) and horizontal composition (*indirect bindings*):
• Vertical composition is resolved automatically, like in original HPE;
  – Binding resolution based on abstract components and contexts;
• Ports between applications are connected by the user;
  – Application #-components have the granularity of CCA components;
• Correction: contradicting [21], CCA frameworks may support recursive composition through AbstractFramework interface;
  – But frameworks still do not provide tools for dealing with such feature;
The features of the CCA framework HPE

Non-Functional Concerns

- One of the distinguishing features of GCM;
- HPE supports kinds for some non-functional concerns;
  - Environments, Qualifiers, and Architectures;
- Controllers are not yet supported
  - “dynamic reconfiguration according to observed performance and failures”;
  - In GCM, controllers are special components (not a regular one);
  - Thus, HPE can include a new kind of #-component (Controllers);
    - BindingController, AttributeController, ContentController, etc ...
The features of the CCA framework HPE

Non-Functional Concerns

controller components

controller ports

regular functional provides ports

regular functional uses ports

C1

... 

C2

... 

S

...
Topics

• Parallelism in component models for HPC;
• The Hash Component Model;
• HPE – Hash Programming Environment;
• Why to make HPE a CCA framework?
• How to make HPE a CCA framework?
• What HPE brings to the world of CCA frameworks?
• Conclusion / Further Works;
Conclusion / Further Works

• With HPE, our group expects to give contributions to CCA community (researchers and users), as well as to improve the communication of novel ideas;

• We need to validate the HPE framework:
  – Test with applications;
    • looking for suggestions and collaborations;
  – Investigate programming techniques;
    • Skeletons;
    • Design and programming of bindings;
  – Working a web-based interface for BuilderService port;
  – Performance evaluation.
The Design of a CCA Framework with Distribution, Parallelism, and Recursive Composition

Francisco Heron de Carvalho Junior
Ricardo Cordeiro Corrêa

Research Group On Parallelism, Graphs, and Optimization
Mestrado e Doutorado em Ciência da Computação (MDCC)
Universidade Federal do Ceará
Fortaleza, Brazil

http://hash-programming-environment.googlecode.com
heron@lia.ufc.br