Introduction to Distributed HTC and overlay systems

Tuesday morning session

Igor Sfiligoi <isfiligoi@ucsd.edu>
University of California San Diego
Logistical reminder

• It is OK to ask questions
  - During the lecture
  - During the demos
  - During the exercises
  - During the breaks

• If I don't know the answer,
  I will find someone who likely does
High Throughput Computing

• Yesterday you were introduced to HTC
  - Often called batch system computing
  - A paradigm that emphasizes maximizing the amount of useful computing over long periods of time
Local HTC

• What you have really experienced so far is **local HTC**

• i.e. computing on dedicated cluster of dedicated resources
  - Managed by a single admin group
  - Co-located in a single location
Is there anything else?

• As you might expect, there are several insulated “local HTC” clusters installed around the world

• And there are non-HTC systems out there, too
Is there anything else?

- As you might expect, there are several **insulated** “local HTC” clusters installed around the world.
- And there are non-HTC systems out there, too.

And you point is ...???
Just local HTC

• You moved from a single PC
  - $O(1)$ cores

• To a local HTC cluster
  - Say, $O(100)$ cores daily avg

Great! I can have my results back in $\frac{1}{100}$th of the time
Just local HTC

- You moved from a single PC
  - O(1) cores
- To a local HTC cluster
  - Say, O(100) cores daily avg
- But is it fast enough?

The result of the 10 body simulation is very promising!
I want to run a 100 body one!

It will still take me over a month to get the results back!
How to get more?

• If you find out that you are resource constrained
  - What do you do?
How to get more?

- If you find out that you are resource constrained
  - What do you do?
- Beg for a larger share of the local pool
  - i.e. better priority compared to the other users of the same pool

There are likely 100s of users

But you better be doing something really important
How to get more?

- If you find out that you are resource constrained
  - What do you do?
- Beg for a larger share of the local pool
- Pay to get more resources bought into the pool
  - Great for long term needs
    - If you can afford it
  - But will not help you in the short term

Getting new machines installed can take months!
How to get more?

• If you find out that you are resource constrained
  - What do you do?
• Beg for a larger share of the local pool
• Pay to get more resources bought
• Get the needed resources somewhere else
  - i.e. not locally
How to get more?

• If you find out that you are resource constrained
  - What do you do?
• Beg for a larger share of the local pool
• Pay to get more resources bought
• Get the needed resources somewhere else
  - i.e. not locally

This is what this lecture is all about!
Distributed HTC

• A computing paradigm that aims at maximizing useful computation using any available resource located anywhere on the planet.

• As a corollary
  - Compute resources are owned and operated by several independent groups.
Implications of distributed computing

• We will be dealing with multiple independent compute systems
  - That do not know about each other

They are owned by several independent groups, remember?
Implications of distributed computing

- We will be dealing with multiple independent compute systems
  - That do not know about each other
- No global HTC scheduler out of the box
Implications of distributed computing

• We will be dealing with multiple independent compute systems
  - That do not know about each other

• No global HTC scheduler out of the box
  - Will have to stitch them all together
The naïve way

• The simplest way is to **partition** your jobs and submit a subset to each cluster.
The naïve way

• The simplest way is to **partition** your jobs and submit a subset to each cluster

• The drawbacks
  - You may need multiple user accounts
  - You may need several different tools
  - Doing a good partitioning is hard
    ▪ Want proportional to the resources you **will** get
  - And what if those CPUs are not in a HTC setup at all?
The naïve way

• The simplest way is to **partition** your jobs and submit a subset to each cluster

• The drawbacks
  - You need mulitple user accounts
  - You need different tools
  - Doing a good partitioning is hard
    ▪ Want proportional to the resources you will get
  - And what if those CPUs are not in a HTC setup at all?

What's the alternative???
Use an overlay system

- Use a systems that looks and feels like a regular HTC to users
  - But has compute nodes all over the world

I like the sound of this!
Use an overlay system

• Use a system that looks and feels like a regular HTC to users
  - But has compute nodes all over the world

But... didn't you just say there was no such thing???
Use an overlay system

• Use a systems that looks and feels like a regular HTC to users
  - But has compute nodes all over the world

  I said “out of the box”
  But one can create one.

  But... Isn't you just say there was no such thing???
Creating a dynamic overlay sys

- No single person cannot manage all the existing resources
Creating a dynamic overlay sys

• But we can lease a subset of them
  - We discuss the how later
Creating a dynamic overlay sys

- But we can lease a subset of them
- And instantiate a HTC system on them
Creating a dynamic overlay sys

- But we can lease a subset of them
- And instantiate a HTC system on them
  - Now we can schedule user jobs on them
  - Only “our” resources are considered
DHTC through an overlay sys

• Just another HTC system
  - Well, almost
  - More details in a few slides

Scheduler

Cool!

Jobs
Overlay system ownership

• Setting up an overlay a major task
  - Comparable with installing a dedicated cluster

• Long term maintenance is also costly

• Not something a final user would want to do
Typical overlay sys operators

- Existing HTC admins, e.g.
  - The UW HTC cluster can “overflow” into OSG
  - The UCSD operates one for local users
- Scientific communities, e.g.
  - The CMS LHC experiment
- The Open Science Grid itself
  - With the OSG Connect

More on this later today
Is DHTC really just HTC?

• Even with overlays, there are some **differences** between DHTC and HTC

• With or without overlays, the core reasons are:
  - Multiple independent HW operators
  - Not all resources are co-located
The multiple owners problem

• In the “Grid” world, the resource owner decides which Operating System, which OS services and which libraries to install
  - A way smaller problem in the “Cloud” world
  - But most of the current DHTC landscape is based on the Grid paradigm
• Different clusters likely configured differently
The multiple owners problem

- In the “Grid” world, the resource owner decides which Operating System, which OS services and which libraries to install
  - A way smaller problem in the “Cloud” world
  - But most of the current DHTC landscape is based on the Grid paradigm
- Different clusters likely configured differently
- Even if you could get your pet library/service installed on some clusters, you cannot expect to get it installed everywhere
Heterogeneity

• DHTC systems are way more heterogeneous than “local HTC” ones

• Two ways to approach this:
  - Minimize external dependencies in your compute jobs
    ▪ Make them self-contained
    ▪ Adapt to the running environment (e.g. multiple binaries, one per platform)
    ▪ Do not use licensed software
Heterogeneity

• DHTC systems are way more heterogeneous than “local HTC” ones

• Two ways to approach this:
  - Minimize external dependencies in your compute jobs
    ▪ Make them self-contained
    ▪ Adapt to the running environment (e.g. multiple binaries, one per platform)
    ▪ Do not use licensed software

Sounds like a lot of work!
Heterogeneity

• DHTC systems are way more heterogeneous than “local HTC” ones.

• Two ways to approach this:
  - Minimize external dependencies
  - Use only a subset of the resources
    ▪ Restrict where your jobs can run
    ▪ Your job will of course take longer to finish
    ▪ May still get you the result sooner
Heterogeneity

- DHTC systems are way more heterogeneous than “local HTC” ones
- Two ways to approach this:
  - Minimize external dependencies
  - Use only a subset of the resources
    - Restrict where your jobs can run
    - Your job will of course take longer to finish
    - May still get you the result sooner

So long for DHTC!
Heterogeneity

- DHTC systems are way more heterogeneous than “local HTC” ones
- Two ways to approach this:
  - Minimize external dependencies
  - Use only a subset of the resources
    - Restrict where your jobs can run
    - Your job will of course take longer to finish
      May still get you the result sooner

Unfortunately, those are the only two alternatives.
No shared file system

• As a side effect, you cannot expect a globally shared file system
  - It's just “yet another user requested service”
• You will have to deal with data explicitly
  - Either using the HTC scheduler capabilities (remember yesterday's lecture)
  - Or, embed file transfer to and from permanent storage into your own jobs
• More details tomorrow
The location problem

- Nodes in different locations need a way to talk to each other
  - This is what networks are for
- If your computation is mostly about CPU cycles, with little input and output data
  - Node location is not an issue at all
- If you have lots of data
  - Remember, throughput is typically inversely proportional with the distance
The data problem

- Transferring large amounts of data over Wide Area Network can take a lot of time
- You should try to compute close to the data source and/or sink
  - Network-wise
  - More about this tomorrow
Bottom line

• For simple computation, DHTC is very similar to HTC
• As soon as you require any complex setup for your jobs, you are in for a rough ride
  - This includes large datasets
Infrastructure considerations

- DHTC is likely to give you access to many more compute slots
- Which is mostly a good thing
  - You get your results faster
- But could crash your HTC system, e.g.
  - Can your storage system handle more data traffic?
  - Can the job scheduling system handle an order of magnitude more nodes?
Infrastructure considerations

- DHTC is likely to give you access to many more compute slots
  - Which is mostly a good thing
    - You get your results faster
  - But could crash your HTC system, e.g.
    - Can your storage system handle more data?
    - Can the job scheduling system handle an order of magnitude more nodes?

Hopefully not something final users should deal with but it is good to keep in mind.
Questions?

- Feel free to ask me questions later:
  Igor Sfiligoi <isfiligoi@ucsd.edu>

- Where to get the needed resources
- glideinWMS – the OSG overlay software
- Hands-on exercises
- Tour
Beware the power

 Courtesy of fanpop.com
Copyright statement

• Some images contained in this presentation are the copyrighted property of ToonClipart.
• As such, these images are being used under a license from said entities, and may not be copied or downloaded without explicit permission from ToonClipart.