The ClassAd Language

ClassAds: The *common language* in HTCondor

ClassAds: 3 uses

Description of entities in Condor

describes machines, jobs, services

Query language to select entities in Condor

“show me all the busy machines”

“show me idle jobs needing > 32 Gb ram”

2 way matching

Given jobs & machines, find matches

ClassAds *describe* all Entities

Entity	How to display full classad
Active Jobs	`$ condor_q -l`
Terminated Jobs	`$ condor_history -l`
Machines (slots)	`$ condor_status -l`
Finished jobs on machine	`$ condor_history -l -file $(condor_config_val STARTD_HISTORY)`
Active submitters	`$ condor_status -submitter -l`
Accounting records	`$ condor_userprio -l`
Schedd service	`$ condor_status -schedd -l`
All services	`$ condor_status -any -l`

ClassAds as *Job* Description

Set of *Attributes*

Attribute:

Key = Value

Key is a name

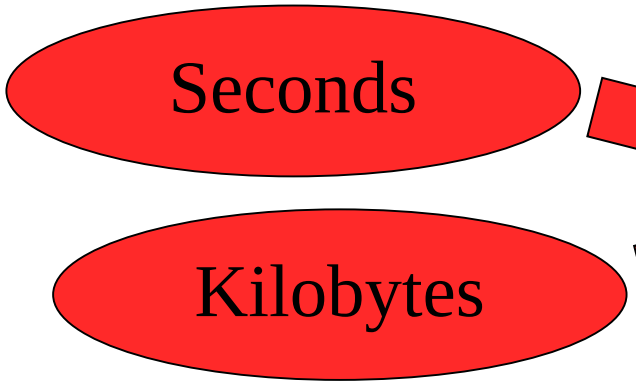
Value has a type

```
$ condor_q -l 180.0
```

```
ClusterId = 180
Cmd = "sleep"
DiskUsage = 100
ExitBySignal = false
QDate = 1535384632
RemoteUserCpu = 12.7
RequestDisk = DiskUsage
… (many attributes removed)
```

ClassAds as *Job* Description

Units by context




```
$ condor_q -l 180.0

ClusterId = 180
Cmd = "sleep"
DiskUsage = 100
[illegible] = false
[illegible]384632
RemoteUserCpu = 12.7
RequestDisk = DiskUsage
… (many attributes removed)
```

Attribute Names (before the =)

- Are like “C” (Python, R,Matlab…) identifiers
  - Must start with letter, then letters, numbers, _
  - No limit on length, but be reasonable
  - Case insensitive, but CamelCase is traditional
  - Extendable – you can add custom attributes
    - (covered in another talk)

Main ClassAd types

Type	Description
Boolean	*true, false*
Integers	*64 bit signed*
Reals	*64 bit IEEE 754 Double*
Strings	*"quoted"*
Reference	Lookup another attribute

Job Ad Example

Value types deduced

Undefined means "Don't Know"

- Includes missing attributes

```
$ condor_q -l 180.0
ClusterId = 180
Cmd = "sleep"
DiskUsage = 100
ExitBySignal = undefined
NiceUser = false
RemoteUserCpu = 12.7
RequestDisk = DiskUsage
… (many attributes removed)
```

ClassAd Expressions

Expressions combine values

C/Java/Python-like:

Logical: *>, <, >=, <=, ==, !=, &&, ||, !*
evaluate to boolean

Math*: +, -, /, *, <<, >>, % evaluate to number*

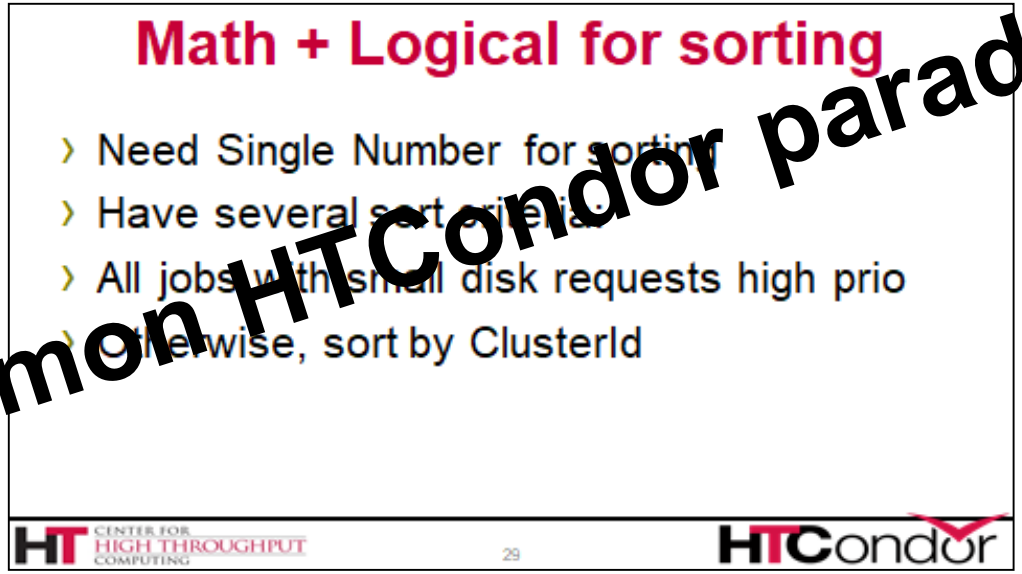
Functions *(builtins)* *depends on function*

Math + Logical for sorting

- Need Single Number for sorting
- Have several sort criteria:
  - All jobs with small disk requests high prio
  - Otherwise, sort by ClusterId

Booleans expand to integers

((DiskUsage < 100) * 1000000) + ClusterId



Common HTCondor paradigm!

ClassAd Builtin Functions

Expression	Returns
time()	Current time in seconds from epoch
substr(str, offset, len)	Extract substring
regexp(pattern, str)	Regexp match (pcre based)
random(x)	Random number from 0 to x
IsUndefined(expr)	True if expr is undefined
StringListMember(s, l)	Is s in list l, where l like "a, b, c"
toUpper(s)	Upper-case s

Control Flow

- Expr ? tExpr : fExpr
  - If expr evals to True, use tExpr, else fExpr
- IfThenElse(expr, tExpr, fExpr)
  - ditto
- (Expr ?: UseThisIfExprWasUndefined)

ClassAd Lists and Nesting

```
childCpus[1]
      -> 2

childCpus[SlotId]
      -> 4

Size(childCpus)
      -> 4
```

```
$ condor_status -l a_pslot
```

```
Name = "fastmachine"
ChildCpus = {1, 2, 3, 4}
slotId = 3


… (many attributes removed)
```

Nested ClassAd

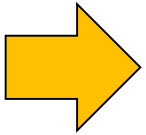
```
childSlot.Name
      -> "slot1"

childSlot.Cpus
      -> 4

childSlot["name"]
      -> "slot1"
```

```
$ condor_status -l a_pslot
```

```
Name = "fastmachine"
ChildSlot =[
      Name = "slot1";
      Cpus = 4
]
Cpus = 40
… (many attributes removed)
```

Now Available in JSON!

```
Name = "fastmachine"
ChildSlot =[
      Name = "slot1";
      Cpus = 4
]
Cpus = 40
ChildCpus = {1, 2, 3, 4}
slotId = 3
```



```
{
 "Name": "fastmachine",
 "ChildSlot": {
   "Name": "slot1”
   "Cpus": 4,
},
 "Cpus": 40,
 "ChildCpus": [
    1, 2, 3, 4 ],
 "slotId": 3
}
```

ClassAds: On to 2nd use

Description of entities in Condor

describes machines, jobs, services

***Query language* to select entities in Condor**

"show me all the busy machines"

"show me idle jobs needing > 32 Gb ram"

2 way matching

Given jobs & machines, find matches

Query language

- Users can write *expressions* as *queries*
- These *select* a subset from a larger set
- If condor evaluates expression to *TRUE*

Query Language example

```
$ condor_status -const 'some classad expr'
```

```
$ condor_q -const 'some classad expr'
```

Example query

```
$ condor_status -const 'Activity == "Busy"'
$ condor_status -const 'Activity != "Busy"'
```

```
MachineName = "Machine1"
Activity = "Busy"
MemoryUsage = 1024
***
MachineName = "Machine2"
Activity = "Idle"
***
MachineName = "Machine3"
Activity = "Busy"
MemoryUsage = 2048
```

Example query

```
$ condor_status -const 'Activity == "Busy"'
$ condor_status -const 'Activity != "Busy"'
```

```
MachineName = "Machine1"
Activity = "Busy"
MemoryUsage = 1024
***
MachineName = "Machine2"
Activity = "Idle"
***
MachineName = "Machine3"
Activity = "Busy"
MemoryUsage = 2048
```

Example query

```
$ condor_status -const 'Activity == "Busy"'
$ condor_status -const 'Activity != "Busy"'
```

```
MachineName = "Machine1"
Activity = "Busy"
MemoryUsage = 1024
***
MachineName = "Machine2"
Activity = "Idle"
***
MachineName = "Machine3"
Activity = "Busy"
MemoryUsage = 2048
```

Example query

```
$ condor_status -const 'MemoryUsage > 2000'
```

```
MachineName = "Machine1"
Activity = "Busy"
MemoryUsage = 1024
***
MachineName = "Machine2"
Activity = "Idle"
***
MachineName = "Machine3"
Activity = "Busy"
MemoryUsage = 2048
```

Example query

```
$ condor_status -const 'MemoryUsage > 2000'
```

```
MachineName = "Machine1"
Activity = "Busy"
MemoryUsage = 1024
***
MachineName = "Machine2"
Activity = "Idle"
***
MachineName = "Machine3"
Activity = "Busy"
MemoryUsage = 2048
```

Strict Equality Operators

- "foo" == undefined -> undefined
- "foo" != undefined -> undefined

- Sometimes you want
- "foo" != undefined to mean false.

Strict Equality Operators

- =?= and =!= are *Strict Equality* comparisons
- And NEVER return undefined:
- "Some String" =?= undefined -> false
- "Some String" =!= undefined -> true
- undefined =?= undefined -> true

ClassAds: 3rd use

Description of entities in Condor

describes machines, jobs, services

Query language to select entities in Condor

“show me all the busy machines”

“show me idle jobs needing > 32 Gb ram”

2 way matching

Given jobs & machines, find matches

Matchmaking

Requires *TWO ads*, returns *true* or *false*

"In the context of ad1 and ad2"

With a selection expression in the

Requirements value of both ads

Commonly used to match jobs and machines

References when matching

```
IsGood = true
RunTime = 123
Name = "Foo"
SomeName = Name
Price = 23.45
Foo = undefined
U = Missing
```

```
IsGood = true
RunTime = 123
Name = "Bar"
Price = 23.45
Foo = undefined
U = Missing
```

What does SomeName return?

References when matching

```
IsGood = true
RunTime = 123
Name = "Foo"
SomeName = Name
Price = 23.45
Foo = undefined
U = Missing
```

```
IsGood = true
RunTime = 123
Name = "Bar"
Price = 23.45
Foo = undefined
U = Missing
```

What does SomeName return?

References when matching

- Ads are checked in order
- Lookup first in the local ad
- Then the other ad
- To force lookup in specific ad, use "My." or "Target." prefix

References when matching

```
IsGood = true
RunTime = 123
Name = "Foo"
SomeName = TARGET.Name
Price = 23.45
Foo = undefined
U = Missing
```

```
IsGood = true
RunTime = 123
Name = "Bar"
Price = 23.45
Foo = undefined
U = Missing
```

What does SomeName return now?

References when matching

```
IsGood = true
RunTime = 123
Name = "Foo"
SomeName = TARGET.Name
Price = 23.45
Foo = undefined
U = Missing
```

```
IsGood = true
RunTime = 123
Name = "Bar"
Price = 23.45
Foo = undefined
U = Missing
```

What does SomeName return now?

For 2 ads to match, both Requirements -> true

- Evaluate Requirements of one, if true
- Evaluate Requirements of other.
- Note My and Target are relative

Job Ad

```
Type  = "Job"
Requirements =
    HasMatlabLicense
      =?= True
Cmd= "/bin/sleep"
Args = "3600"
Owner = "gthain"
NumJobStarts = 8
```

Slot Ad

```
Type = "Machine"
Cpus  = 40
Memory = 2048
Requirements =
  (Owner == "gthain")   &&
  (TARGET.NumJobStarts <=
   MY.MaxTries)
HasMatlabLicense = true
MaxTries = 4
```

Questions?

Thank You!