Dynamic Slot Tutorial

Condor Project
Computer Sciences Department
University of Wisconsin-Madison

Outline

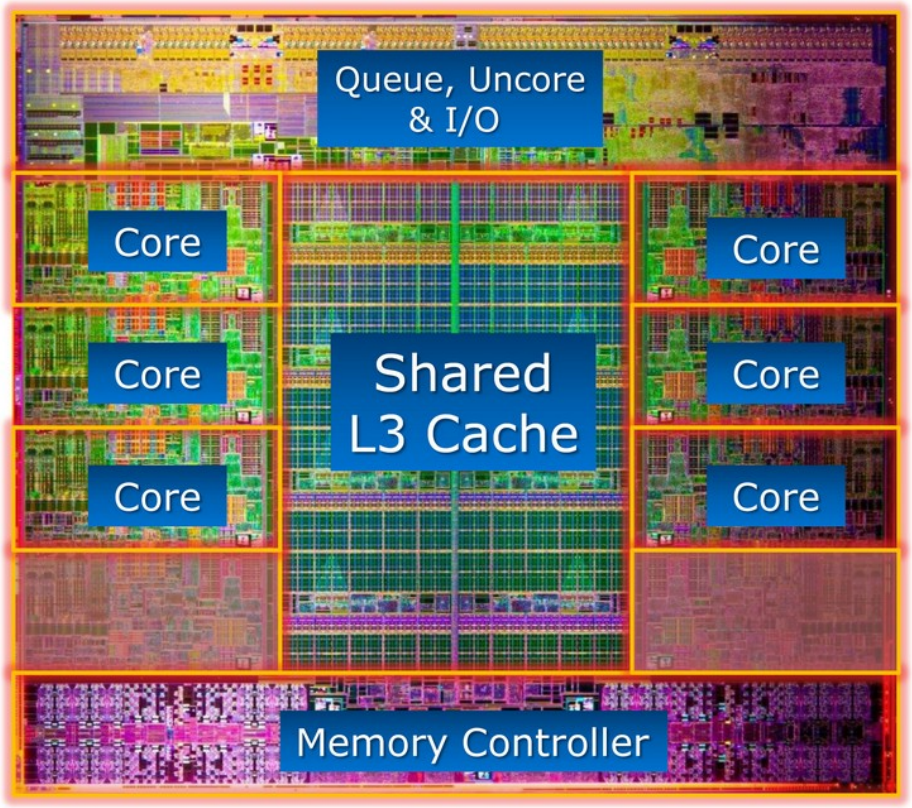
Why we need partitionable slots

How they've worked since 7.2

What's new in 7.8

What's still left to do

What's the Problem?



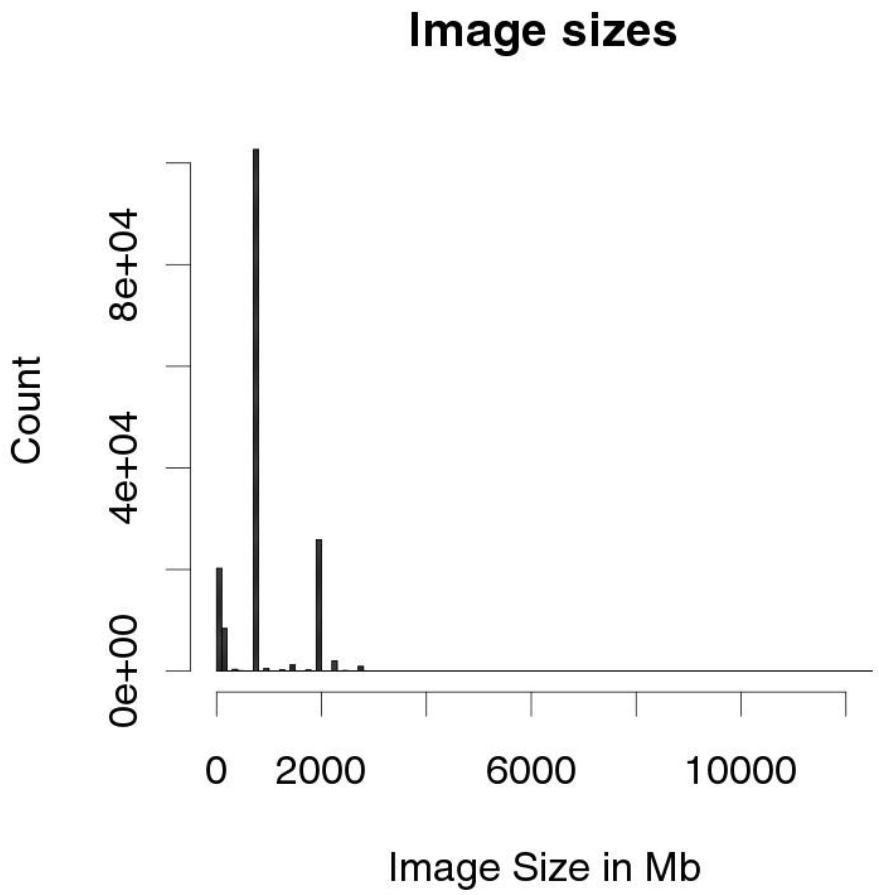
Example Machine:

- 8 cores
- 8 Gigabytes memory
- 2 disks

The old way (Still the default)

```
> $ condor_status

Name                OpSys      Arch   State     Activity LoadAv Mem   ActvtyTime

slot1@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.110  1024  0+00:45:04
slot2@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:05
slot3@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:06
slot4@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:07
slot5@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:08
slot6@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:09
slot7@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:10
slot8@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000  1024  0+00:45:03
                     Total Owner Claimed Unclaimed Matched Preempting Backfill
        X86_64/LINUX     8     0       0         8       0          0        0
               Total     8     0       0         8       0          0        0
```

Problem: Job Image Size



Simple solution: Static non-uniform memory

```
# condor_config
NUM_SLOTS_TYPE_1 = 1
NUM_SLOTS_TYPE_2 = 7
SLOT_TYPE_1 = mem=4096
SLOT_TYPE_2 = mem=auto
```

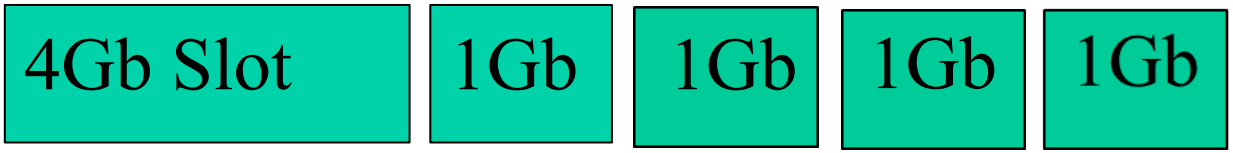
Result is

```
$ condor_status

Name                OpSys      Arch   State     Activity LoadAv Mem
ActvtyTime

slot1@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.150  4096
slot2@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585
slot3@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585
slot4@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585
slot5@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585
slot6@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585
slot7@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585
slot8@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000   585

                     Total Owner Claimed Unclaimed Matched Preempting
Backfill

        X86_64/LINUX     8     0       0         8       0          0
```

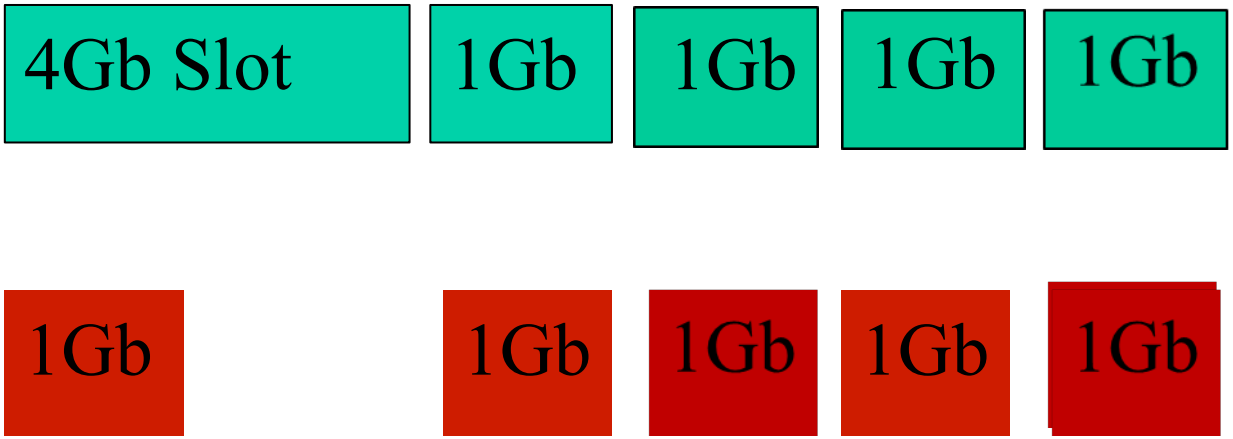
Better, still not good

- Job requirements:
  - Requirements = memory > 2048
- How to steer small jobs to small slot?
  - Trivia question in classads?
- How to pick correct sizes?
- Changes require startd restarts

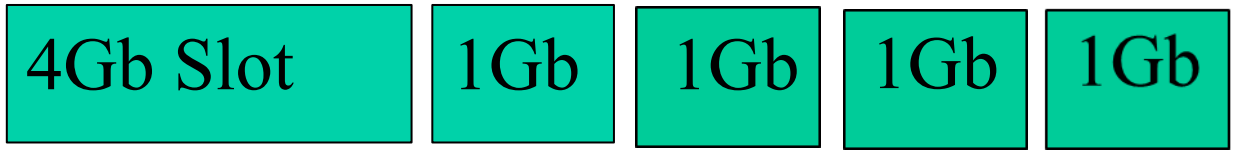
8 Gb machine partitioned into 5 slots

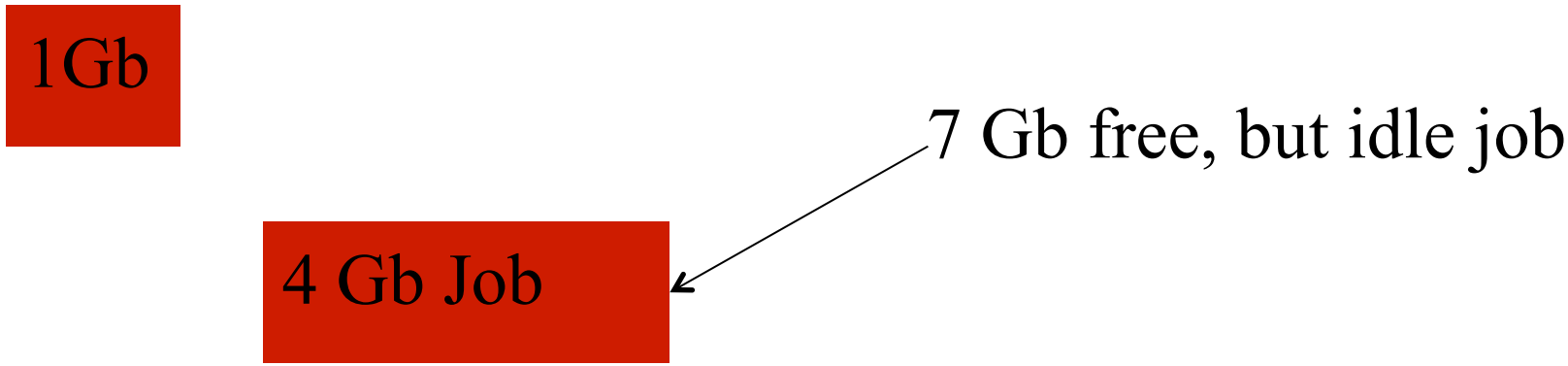


8 Gb machine partitioned into 5 slots



8 Gb machine partitioned into 5 slots





New: Partitionable slots

- Work in progress
- First landed in 7.2
- More work in 7.8
- Even more goodness to come…

- But very usable now

The big idea

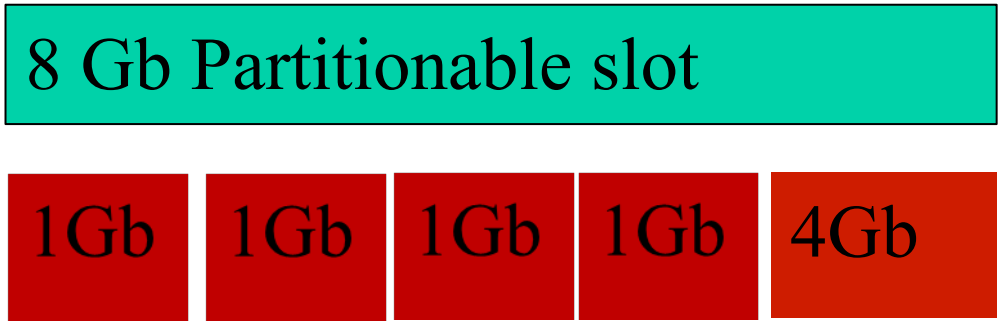
- One "partionable" slot
- From which "dynamic" slots are made
- When dynamic slot exit, merged back into "partionable"
- Split happens at claim time

(cont)

- Partionable slots split on
  - Cpu
  - Disk
  - Memory
  - (Maybe more later)
- When you are out of one, you're out of slots

3 types of slots

- Static (e.g. the usual kind)
- Partitionable (e.g. leftovers)
- Dynamic (usableable ones)
  - Dynamically created
  - But once created, static


8 Gb Partitionable slot
1Gb
1Gb
1Gb
1Gb
4Gb


8 Gb Partitionable slot
1Gb
1Gb
1Gb
5Gb

How to configure

```
NUM_SLOTS = 1
NUM_SLOTS_TYPE_1 = 1
SLOT_TYPE_1 =  cpus=100%
SLOT_TYPE_1_PARTITIONABLE = true
```

Looks like

```
$ condor_status

Name                OpSys      Arch   State     Activity LoadAv
Mem   ActvtyTime

slot1@c LINUX       X86_64 Unclaimed Idle      0.110   8192

                     Total Owner Claimed Unclaimed Matched
       X86_64/LINUX      1     0       0         1       0
              Total      1     0       0         1       0
```

When running

```
$ condor_status

Name                OpSys      Arch   State     Activity LoadAv
Mem   ActvtyTime

slot1@c LINUX       X86_64 Unclaimed Idle      0.110  4096
slot1_1@c LINUX     X86_64 Claimed   Busy      0.000  1024
slot1_2@c LINUX     X86_64 Claimed   Busy      0.000  2048
slot1_3@c LINUX     X86_64 Claimed   Busy      0.000  1024
```

All this in 7.2

- What are the problems?
  - Slow matching
  - Broken for parallel universe
  - Dedicated slots users broken
  - Fragmentation
  - Selection of dynamic slots sizes tricky

Fixed in 7.8

- Matching faster
  - CLAIM_PARTITIONABLE_LEFTOVERS
    - = false to make slow again..
- Parallel universe fixed
- Dedicated slot users fixed
- Support for defragging!

Answer to triva question

- Requirements = (Memory > 1024)
  - How can startd parse?

- It Can't!

THIS IS BIG!

- Memory requirements deprecated
- Don't do
  - Requirements = memory > 1024
- Generates a warning now:

```
condor_submit submit7
Submitting job(s)…
WARNING: your Requirements expression refers to
TARGET.Memory. This is obsolete. Set request_memory
and condor_submit will modify the Requirements
expression as needed.
```

Instead, use

- request_memory = 2048 # mbytes

- Same is true for disk, cpus
- request_disk = 16384 # kbytes
- request_cpus = 1
- Requirements automatically fixed

I have to change all my submit files?

- There's a knob for that ™
- JOB_DEFAULT_REQUESTMEMORY
- JOB_DEFAULT_REQUESTDISK
- JOB_DEFAULT_REQUESTCPUS
- submit side defaults
- Can be expressions…

I don't want to change the config file

JOB_DEFAULT_REQUEST_MEMORY

ifthenelse(MemoryUsage =!= UNDEF, Memoryusage, 1)

JOB_DEFAULT_REQUEST_CPUS

1

JOB_DEFAULT_REQUEST_DISK

DiskUsage

What about the startd side?

- Startd has a say, too:

  MODIFY_REQUEST_EXPR_REQUESTCPUS

  quantize(RequestCpus, {1})

  MODIFY_REQUEST_EXPR_REQUESTMEMORY

  quantize(RequestMemory, {TotalSlotMem/TotalSlotCpus / 4})

  MODIFY_REQUEST_EXPR_REQUESTDISK

  quantize(RequestDisk, {1024})

Why quantize?

- Allow slot reuse

```
Name               OpSys     Arch   State     Activity LoadAv Mem

slot1@c LINUX      X86_64 Unclaimed Idle     0.110  4096
slot1_1@c LINUX    X86_64 Claimed   Busy     0.000  2048
slot1_2@c LINUX    X86_64 Claimed   Busy     0.000  1024
slot1_3@c LINUX    X86_64 Claimed   Busy     0.000  1024
```

Also holds for cpus

- Much easier way to do whole machine
- Basically same as memory requirements
- Easier to set up than Parallel universe

Fragmentation

```
Name     OpSys       Arch    State      Activity LoadAv Mem

slot1@c LINUX       X86_64 Unclaimed Idle      0.110  4096
slot1_1@c LINUX     X86_64 Claimed   Busy      0.000  2048
slot1_2@c LINUX     X86_64 Claimed   Busy      0.000  1024
slot1_3@c LINUX     X86_64 Claimed   Busy      0.000  1024
```

Now I submit a job that needs 8G – what happens?

Solution: New Daemon

- condor-defrag (new in 7.8)
  - One daemon defrags whole pool
    - Central manager good place to run
- Scan pool,try to fully defrag some startds
- Only looks at partitionable machines
- Admin picks some % of pool that can be "whole"

Oh, we got knobs…

DEFRAG_DRAINING_MACHINES_PER_HOUR

default is 0

DEFRAG_MAX_WHOLE_MACHINES

default is -1

DEFRAG_SCHEDULE

- graceful (obey MaxJobRetirementTime, default)
- quick (obey MachineMaxVacateTime)
- fast (hard-killed immediately)

Defrag vs. Preemption

- Defrag can be general purpose
  - Looks only at startds, not at demand
  - Can also preempt non-partitionable slots
    - (if so configured)
- Negotiator preemption looks at 2 jobs

Advanced Topics

- More than one Partitionable slot
- Mix and match partionable slots
- Overcommitting partionable sltos
- Parallel universe

Future work

- Claiming partitionable slots
  - So RANK based preemption works
- condor_q -analyze
- More knobs!

Thank you