GlideinWMS - Bug #11876

rounding for multicore jobs on multicore entries causes less pressure than there should be

03/03/2016 11:58 AM - Marco Mambelli

Status:	New	**Start date:**	03/03/2016
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
First Occurred:		**Stakeholders:**	
Occurs In:			

Description

When

From the code:

```
prop_cpus = (out_cpu_counts[site] * new_out_counts[site_index])/out_glidein_counts[site]
prop_out_count = prop_cpus/glidein_cpus
final_out_cpu_counts[site] = math.ceil(prop_out_count)
```

Which translated in a single formula is, for each “site” (= frontend, entry, group):
ceil ((# of CPUs requested * # glideins assigned) / (# glideins that were idle * GLIDEIN_CPUS))
where # of CPUs requested = requested_cpus * # idle jobs (for each cluster of jobs)

e.g. 100 idle jobs asking 3 cores in a cluster with 4 cores per glidein is reduced to 75 idle jobs requests.

The problem in this re-scaling is that if a non integer # of jobs fit at the site, this is not considered but you cannot split a job between 2 glideins (in other words: you cannot fit 1.5 jobs in a glidein). If there is only one job cluster the ratio should be brought outside the calculation, something like:
ceil ((# idle jobs * # glideins assigned) / (floor(GLIDEIN_CPUS/requested_cpus) * # glideins that were idle))

In a normal situation there are multiple job clusters each requesting a different amount of CPUs split across multiple entries.
To correctly calculate the re-scaling instead of calculating the sum (# of CPUs requested), the request from the job clusters should be kept as list of tuples (# idle jobs, # cores) and the calculation should become:
ceil (sum(# idle jobs / floor(GLIDEIN_CPUS/requested_cpus)) * # glideins assigned / # glideins that were idle)

This affects only multicore jobs, for single core floor(GLIDEIN_CPUS/1) == GLIDEIN_CPUS.
Note that GLIDEIN_CPUS must be known to do this rescaling, otherwise (auto/slot) 1 core is assumed and multicore jobs will not even match.

This is connected in part to #11854