Resource Management and Green Computing

Mehdi Sheikhalishahi

University of Calabria

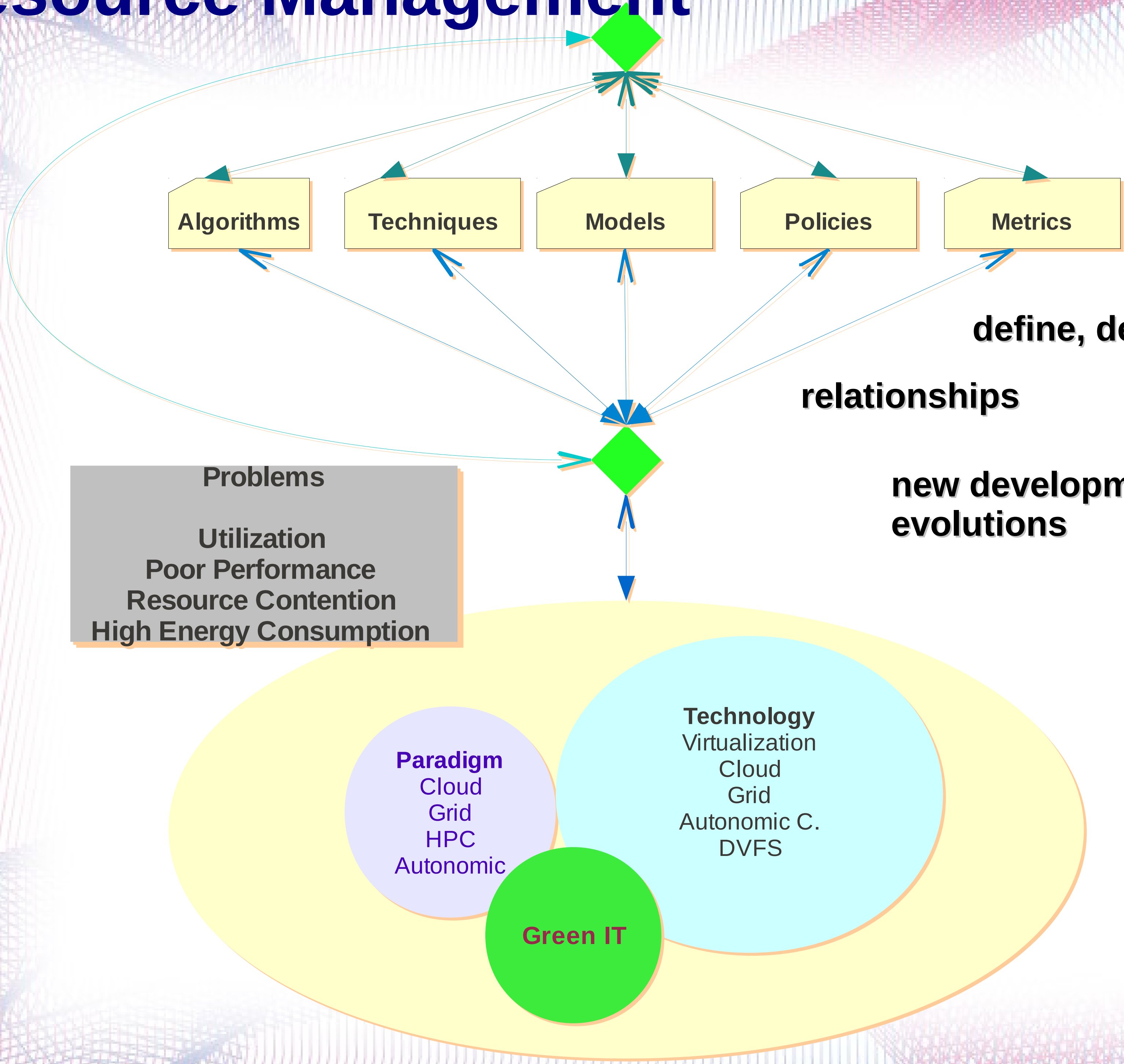
International Research Workshop On Advanced High Performance Computing Systems

Cetraro (Italy), June 27 - 29, 2011

Outline

- Relationship in Resource Management
- Resource Contention Metric for HPC Workloads
- Energy Efficient Consolidation Policies
- Experimental Results
- Conclusion

Resource Management

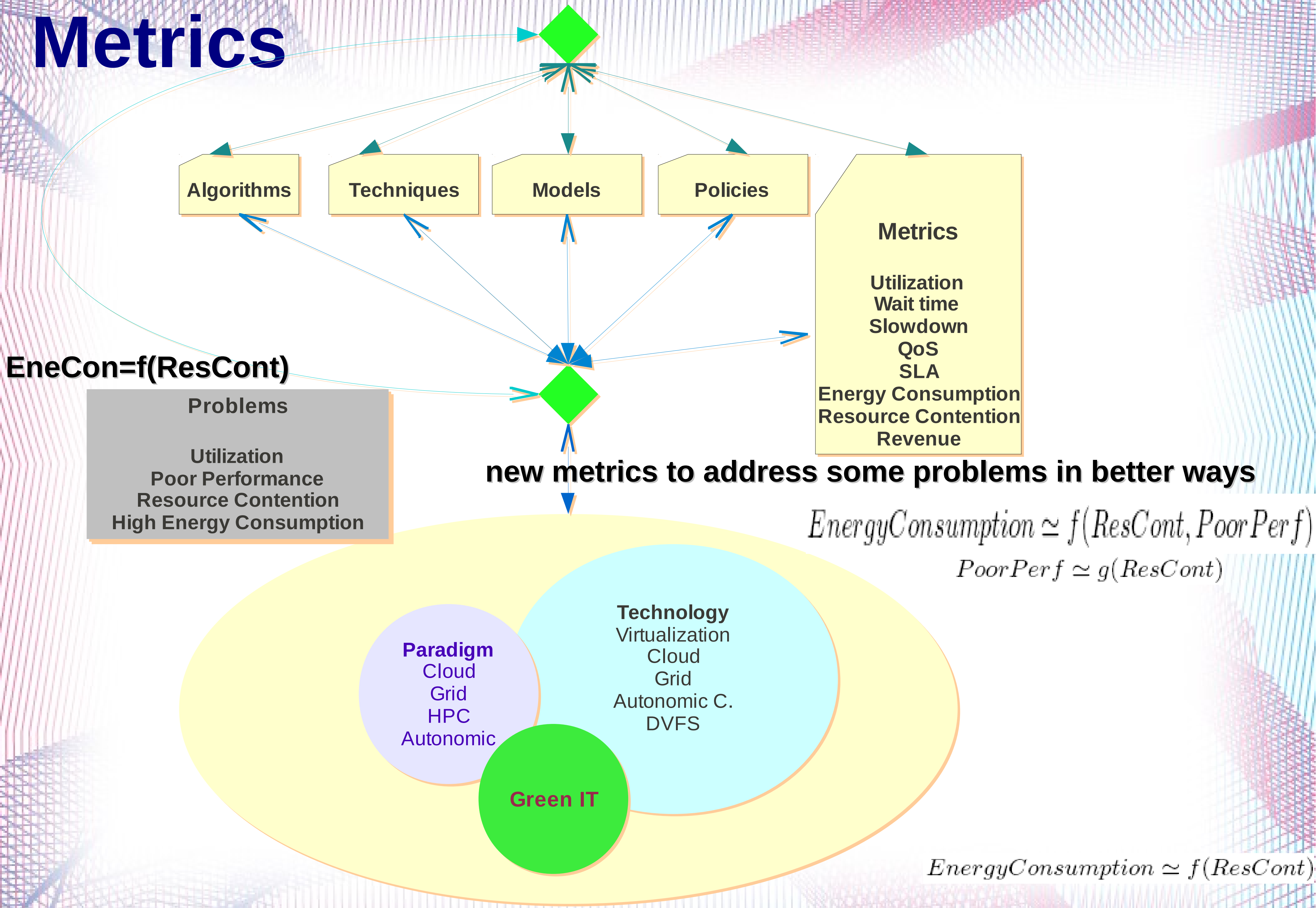


define, design and develop

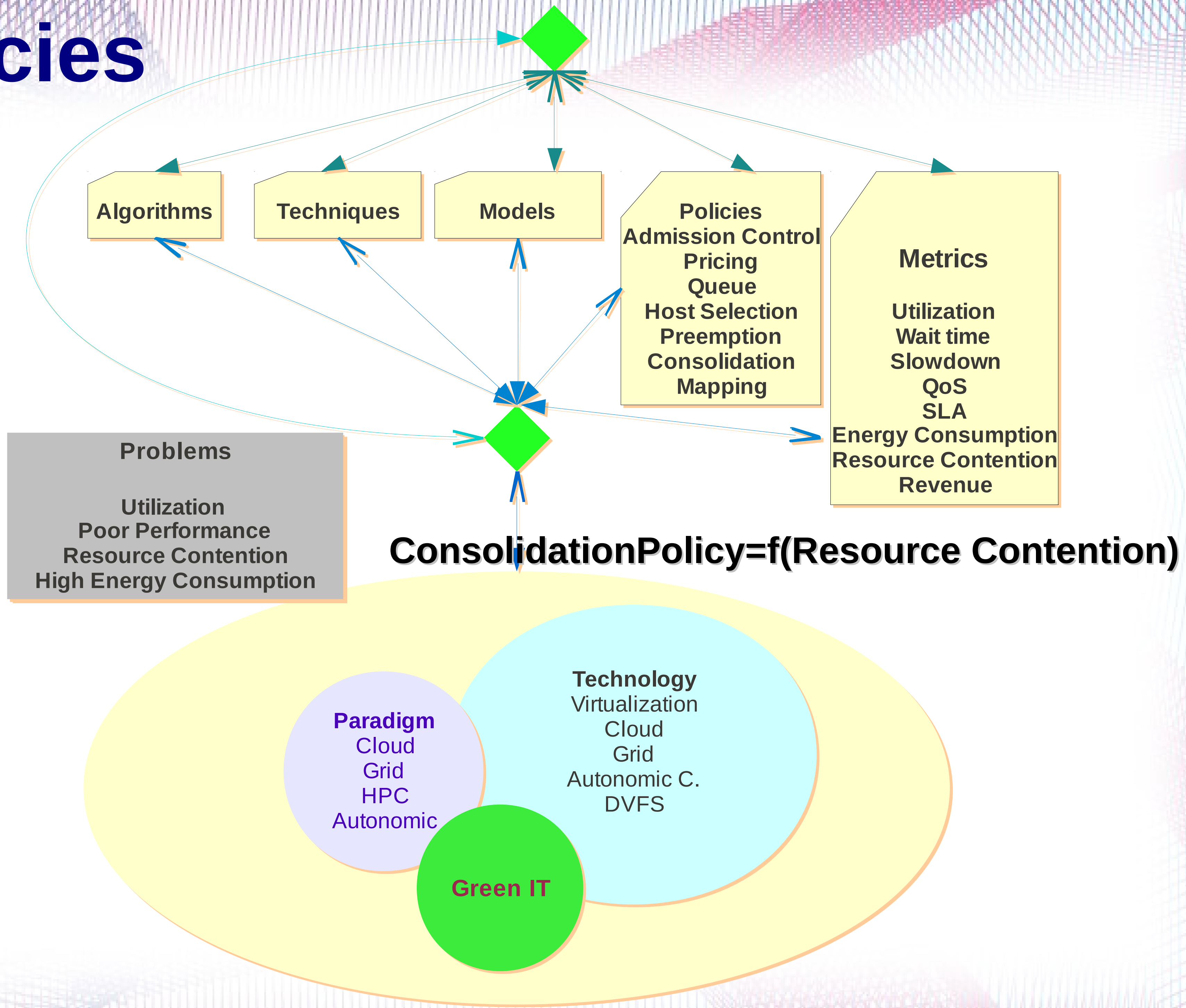
relationships

new developments and evolutions

Metrics


Algorithms
Techniques
Models
Policies
Metrics
Utilization
Wait time
Slowdown
QoS
SLA
Energy Consumption
Resource Contention
Revenue
EneCon=f(ResCont)
Problems
Utilization
Poor Performance
Resource Contention
High Energy Consumption
new metrics to address some problems in better ways
$EnergyConsumption \simeq f(ResCont, PoorPerf)$
$PoorPerf \simeq g(ResCont)$
Technology
Virtualization
Cloud
Grid
Autonomic C.
DVFS
Paradigm
Cloud
Grid
HPC
Autonomic
Green IT
$EnergyConsumption \simeq f(ResCont)$

Policies



Algorithms
Techniques
Models
Policies
Admission Control
Pricing
Queue
Host Selection
Preemption
Consolidation
Mapping
Metrics
Utilization
Wait time
Slowdown
QoS
SLA
Energy Consumption
Resource Contention
Revenue
Problems
Utilization
Poor Performance
Resource Contention
High Energy Consumption
ConsolidationPolicy=f(Resource Contention)
Technology
Virtualization
Cloud
Grid
Autonomic C.
DVFS
Paradigm
Cloud
Grid
HPC
Autonomic
Green IT

Resource Contention Metric for HPC Workloads

- Resource Contention Def.
- Architecture, infrastructure dependent
- HPC Workloads as a case study
- Unit: quantitative

Table 1. HPC Application Characteristics

HPC Char./Resource	CPU	Cache	Memory	IO	Net-in	Net-out
Compute-intensive	✓	✓	✓	✓	✓	✓
Data-intensive				✓	✓	✓
Memory-intensive		✓	✓		✓	✓
Comm-intensive				✓	✓	✓

Resource Contention Metric for HPC Workloads

$$RC(t) = \sum_{\forall n \in PhyNodes} RC(t,n) \qquad RC(t,n) = \sum_{\forall r \in resTypes} RC(t,n,r)$$

Let $JobsSchedOn(n,t)$ be the scheduled jobs on physical host n at time t

$$JS(t,n) = JobsSchedOn(n,t)$$

Let $j.stresson$ be resources which job j puts stress on them

Let $j.resReq$ be resource requirements of job j in terms of capacity

$$resContFlg(t,n,r) = \begin{cases} 1, \sum_{\forall j \in JS(t,n)} (r \in j.stresson == True?1:0) > 1 \\ 0, Otherwise \end{cases}$$

$$RC(t,n,r) = \begin{cases} \sum_{\forall j \in JS(t,n) \wedge r \in j.stresson} j.resReq[r], & resContFlg(t,n,r) = 1 \\ 0, & Otherwise \end{cases}$$

Effective Energy Aware Consolidation Policies

- Effective placement of jobs
- Two effective energy aware consolidation policies in terms of resource contention model
- Host selection policies of Haizea as a framework: score based system

EA Consolidation Base Model

$$Score(Job\, j, Time\, t, Node\, n) = \sum_{\forall r \in j.stresson} ScoreRes(r)$$

$$\forall r \in j.stresson\; ScoreRes(r) = \begin{cases} \sum_{\forall js \in JS(t,n) \wedge r \in js.stresson} js.resReq[r], & resContFlg(r) = 1 \\ 0, & Otherwise \end{cases}$$

$$\forall r \in j.stresson\; resContFlg(r) = \begin{cases} 1, (\sum_{\forall js \in JS(t,n)} (r \in js.stresson == True ? 1 : 0)) \geq 1 \\ 0, Otherwise \end{cases}$$

Simple Policy

Algorithm 2 NodeScoreAtTimeT(Job j, Time t, Node n)

$Score \Leftarrow 0$
for all r such that $r \in j.stresson$ **do**
  $Flag \Leftarrow False$
  for all js such that $js \in JS(t, n)$ **do**
    if $r \in js.stresson$ **then**
      $Flag \Leftarrow True$
      break
    end if
  end for
  if $Flag = True$ **then**
    for all js such that $js \in JS(t, n)$ **do**
      if $r \in js.stresson$ **then**
        $Score \Leftarrow Score + js.resReq[r]$
      end if
    end for
  end if
end for
return $Score$

Simple Policy

Algorithm 1 EAConsolidationPolicy(Job j, Time t)

$NodesScore \Leftarrow empty\ dictionary$

for all n such that $n \in PhyNodes$ **do**

  $Score \Leftarrow NodeScoreAtTimeT(Job\ j, Time\ t, Node\ n)$

  $NodesScore[n] \Leftarrow Score$

end for

return $Sort\ NodesScore\ according\ to\ values\ and\ return\ keys\ in\ order$

Consolidation Policy Over Job Time Horizon

Algorithm 3 EATimeConsolidationPolicy(Job j, Time t)

```
NodesScore ⇐ empty dictionary
jobDuration ⇐ j.duration
for all n such that n ∈ PhyNodes do
   Score ⇐ 0
   time ⇐ t
   changePoints ⇐ getChangePointsAfter(Time = time, until =
   jobDuration, node = n)
   if len(changePoints) > 0 then
      while chp ⇐ changePoints.getNext() do
         tsDuration ⇐ chp − time
         Score ⇐ Score + NodeScoreAtTimeT(j, time, n) ∗ tsDuration
         time ⇐ chp
      end while
      if time < t + jobDuration then
         Score ⇐ Score + NodeScoreAtTimeT(j, time, n) ∗ (t + jobDuration −
         time)
      end if
      Score ⇐ Score/jobDuration
   else
      Score ⇐ NodeScoreAtTimeT(j, time, n)
   end if
   NodesScore[n] ⇐ Score
end for
return Sort NodesScore according to values and return keys in order
```

Experimentations

Commodity cluster infra. as resource model

Workload archives from Parallel Workloads Archive

We synthetically generate job attributes by uniform distributions

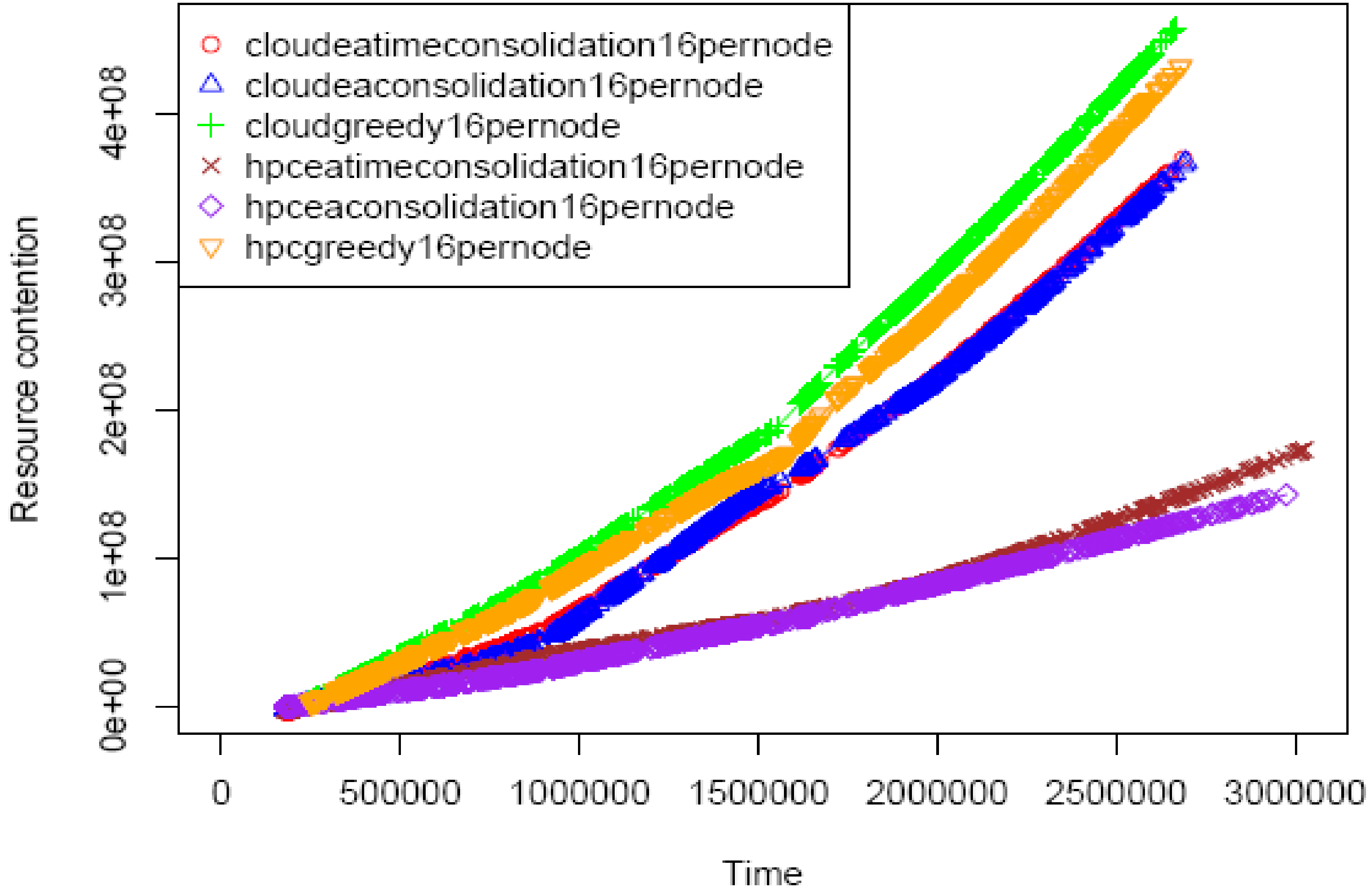
Table 2. HPC Characteristics distribution in workload trace

HPC Char.:Prob./Resource	CPU Memory	CPU	Memory	IO	Net-in Net-out	Net-in	Net-out
Compute-intensive: $\frac{1}{2}$	$\frac{6}{10}$	$\frac{4}{10}$	0	$\frac{1}{2}$	$\frac{4}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
Data-intensive: $\frac{1}{6}$	$\frac{3}{10}$	0	$\frac{7}{10}$	$\frac{1}{2}$	$\frac{4}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
Memory-intensive: $\frac{1}{6}$	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{4}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
Comm-intensive: $\frac{1}{6}$	0	0	0	$\frac{1}{2}$	1	0	0

Configurations

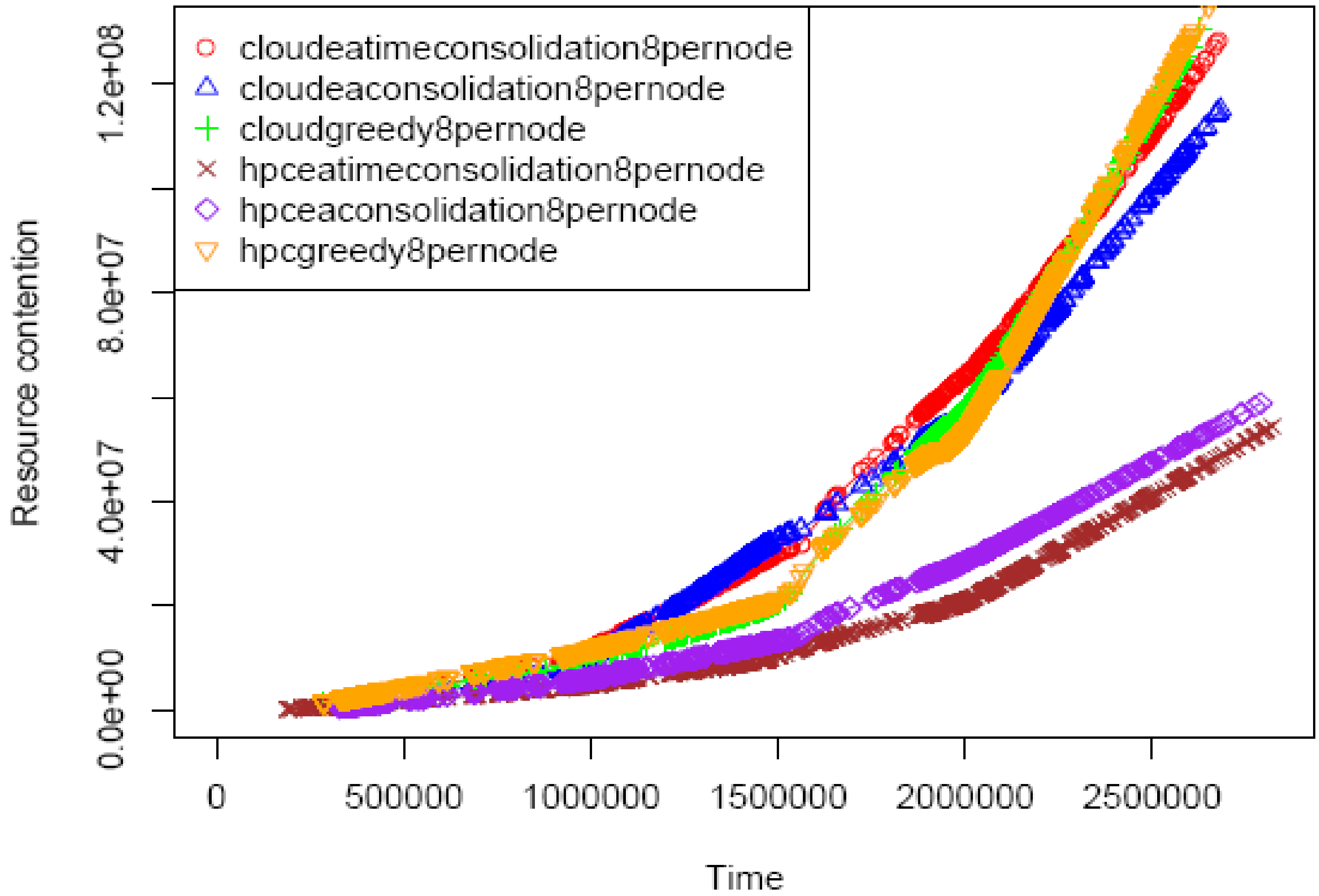
- SDSC Blue Horizon from the Parallel Workloads Archive
- Computing Paradigms: cloud, HPC
- Consolidation Policies: greedy, SimpleEAConsolidation, EAConsolidationOverJobTimeHorizon
- Sites: With multi-instance type CPU 8, 16 number of cores per physical nodes
- Backfilling: conservative

Results(16 cores)



(a) 16 cores

Results(8 cores)



(b) 8 cores

Final Results

Table 3. Completion Time and Resource Contention metric for 16 and 8 cores

Configuration	Time	Resource Contention
cloudeatimeconsolidation	2689207	365667584
cloudeaconsolidation	2694286	372760333
cloudgreedy	2663556	455026120
hpceatimeconsolidation	3058147	201649827
hpceaconsolidation	3019506	170207159
hpcgreedy	2681813	441726388

(a) 16 cores

Configuration	Time	Resource Contention
cloudeatimeconsolidation	2681514	128798324
cloudeaconsolidation	2682099	114294790
cloudgreedy	2663556	125467388
hpceatimeconsolidation	2833329	54352992
hpceaconsolidation	2840499	60079428
hpcgreedy	2672287	129578192

(b) 8 cores

Conclusions

- greedy vs. energy aware policies: Almost in all cases energy aware policies outperform greedy policies
- SimpleEAConsolidation vs. EAConsolidationOverJobTimeHorizon: there is no clear outperformance of the one over the other
- cloud vs. HPC: In spite of the fact that cloud scheduling is more precise than HPC; surprisingly we observe that HPC results are much better than cloud's. This result is the same as in-depth research on the impact of inaccurate estimates on pure scheduling metrics

Thanks

?