Modeling Resilience in Cloud-Scale Data Centers

John Cartlidge
School of Computer Science
University of Bristol, UK
john.cartlidge@bristol.ac.uk

The work presented here was performed with Ilango Sriram at the University of Bristol, UK, and was funded by HP-Labs and EPSRC grant EP/H042644/1 as part of the national Large-Scale Complex IT Systems (LSCITS) initiative headed by Professor Dave Cliff.

Outline

- Background: Cloud Computing
- Problem: Ultra-large scale data centres (DCs) in production with insufficient pre-testing
- Long-Term Research Aim: To create a robust simulation modelling framework for cloud DCs
- Today: Modelling failure resilience
- Demonstrate resilience and efficiency of different redundancy scheduling algorithms

Cloud Computing: "the next step"

The commercial provision of IT has undergone "step" changes every 10 years or so…

1960s: **Mainframes** - physically huge, housed in air-con rooms

Early 70s: **Mini-computers** - more robust, compact, affordable

Late 70s: **PCs -** cheaper & smaller, single-user

1980s: **Communication LAN**s: Client-server model first used. Sun's slogan: "the network ***is*** the computer"

Mid 90s: Widespread adoption of the **Internet**

Now: **Cloud**: online provision of centralised utility computing

Computing as a Utility

19th Century: Electricity in Manufacturing

- Electricity generation as important to manufacturing as the factory itself, with each factory having its own generator
- Economies of scale by utility providers made local factory-generation uneconomic

Today: Computing in Business and the Home

- IT is a necessity, but no longer offers any particular advantage since hardware and software is ubiquitous and standardized
- IT can be provided remotely "in the cloud", making in-house data centres uneconomic

Ultra-large “cloud-scale” data centres

- Cloud computing offers the provision of IT resources in every home and business “as a Service”
  - Infrastructure (IaaS), Platform (PaaS), Software (SaaS)
  - Enables companies to focus on what they *should* be doing (their core business), rather than running data-centres
- Providing cloud services requires ultra-large DCs
  - Tens of thousands of servers, and growing …
- Unlike HPC, cloud providers use commodity hardware
  - Advantage comes from scale-out (massive parallelisation) not scale-up (increasing power and performance)

“Normal” Failure

“an application running across thousands of machines may need to react to failure conditions on an hourly basis” (Barosso & Hölzle, Google, 2009)

- In a DC with 300,000 servers, each with an average life of 3 years, we expect +10 deaths / hour
- Failures and outages in such large complex systems are *normal*: not unexpected or rare
- However, for the cloud to work, providers must guarantee availability (often “triple 9”, 99.9%, or more)
- Cloud providers need to build **resilience** into their systems

Where is the (simulation) model?

- Many mature engineering fields have robust industry-standard simulation frameworks for design and testing
  - E.g., SPICE for integrated circuit design, CFD for automotive
- For cloud-scale DC design no simulation model exists
- DCs are going into service without sufficient pre-testing
- At UoBristol we are attempting to build a set of simulation tools for the design and testing of cloud DCs
- Here, we present a preliminary model of resilience

Redundancy Scheduling for Resilience

- Redundancy can be used to counter-act hardware and software failures
  - Multiple instances running in parallel
- However, redundancy is costly
  - Increased computation and network communication

Data Centre Design

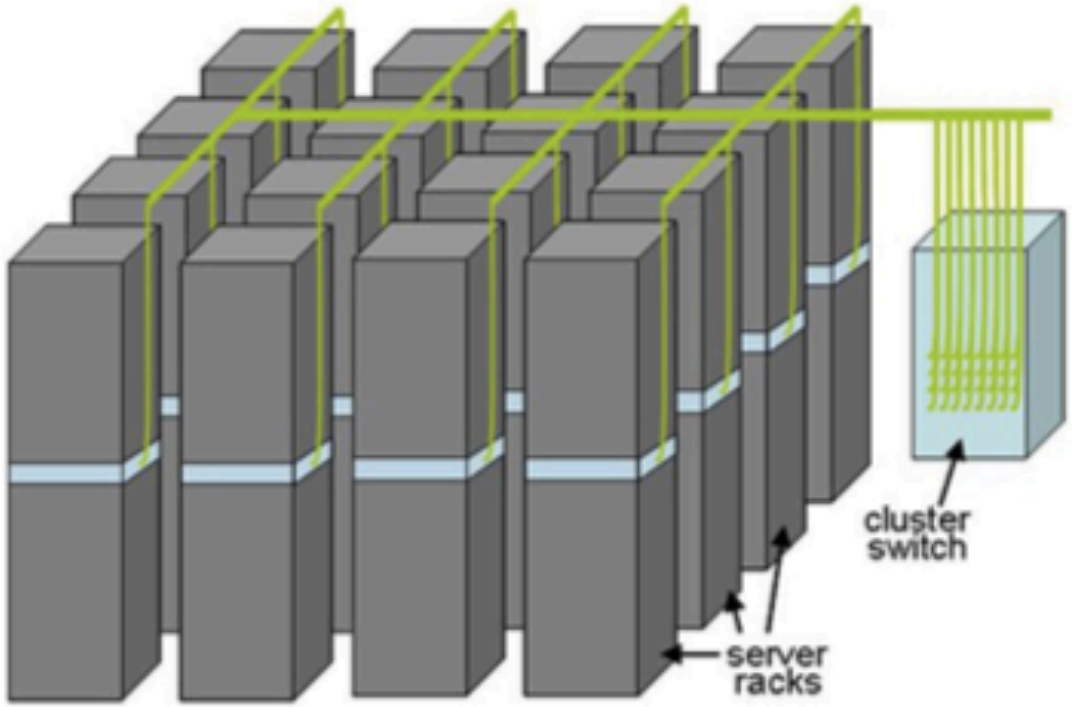
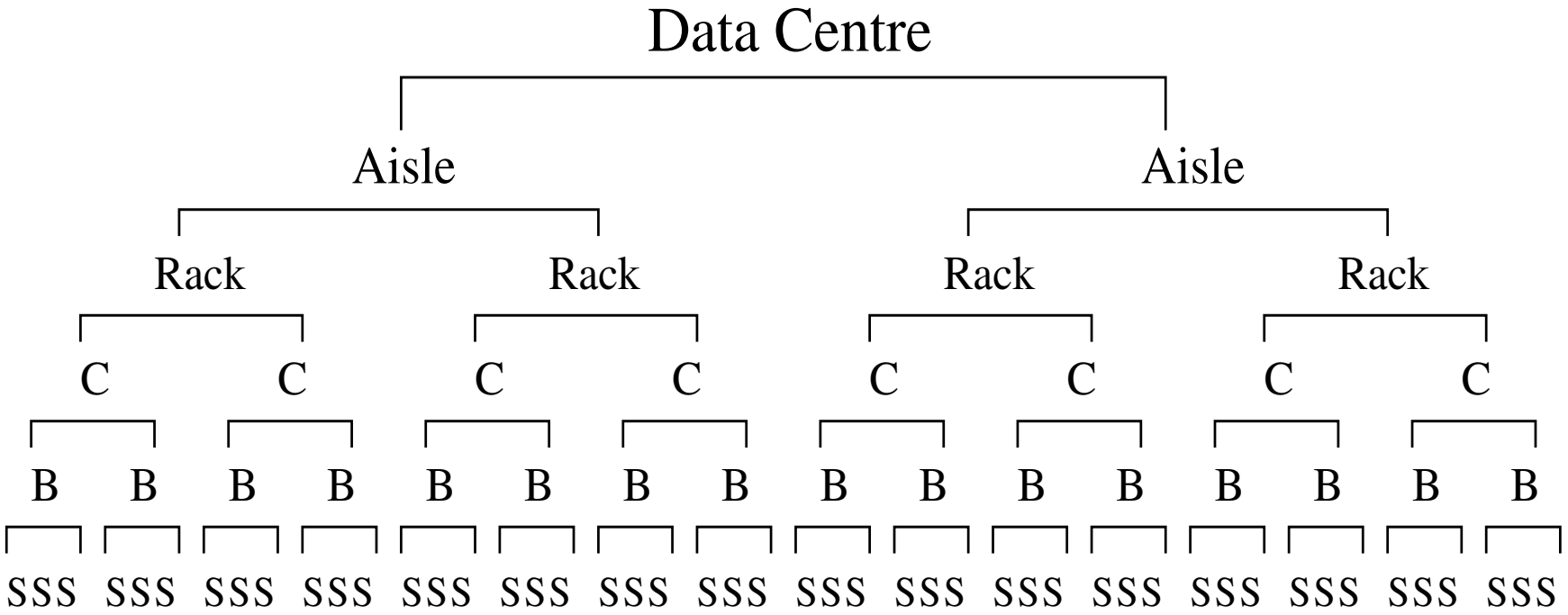


Diagram of a small cluster with a cluster-level Ethernet switch/router. From Barosso & Hölzle, Google, 2009.

Data Centre Model

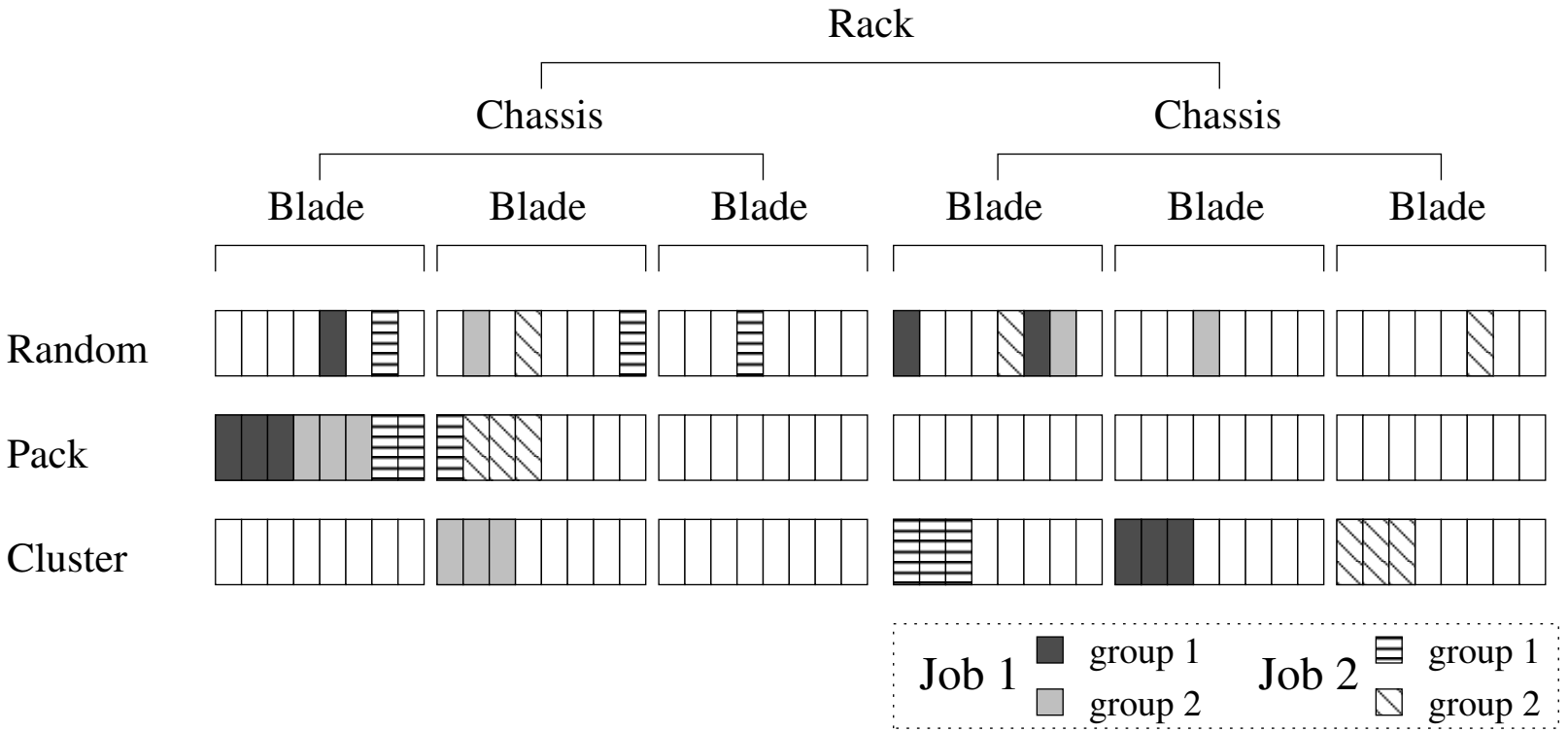


Hierarchical network-tree model of DC. Cloud **S**ervices run on **B**lade servers, which are mounted on **C**hassis in Racks, arranged in Aisles within a DC

Model Assumptions

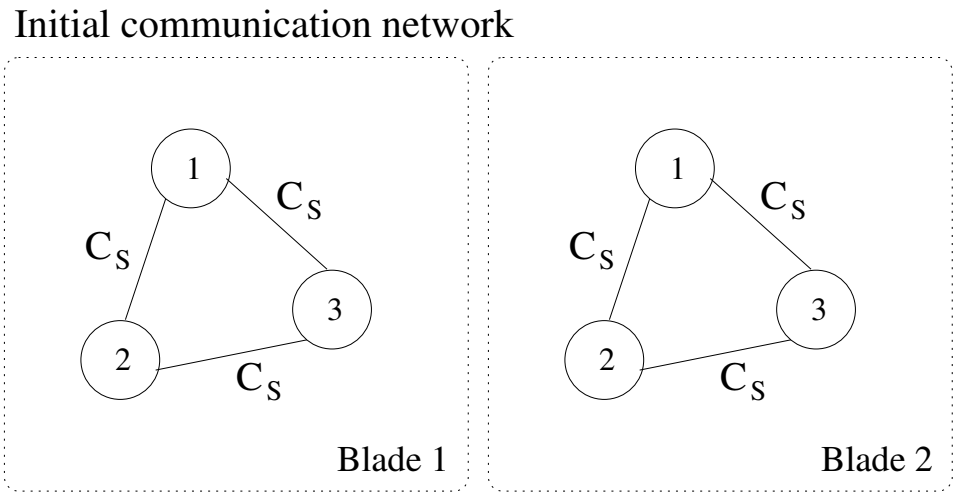
- Failure can occur at any level in the hierarchy tree
- Network costs are greater when "higher" in the tree
  - Bandwidth and latency is greater between racks/aisles than between services running on the same server
- Jobs consist of a set of parallelizable tasks
- Fixed DC size with < 100% utilisation
- Tasks are scheduled using 3 simple algorithms:
  - **Random** (distribute tasks randomly across DC)
  - **Pack** (pack all tasks into the smallest region possible)
  - **Cluster** (pack all tasks within a redundancy group together, distribute groups randomly across the DC)

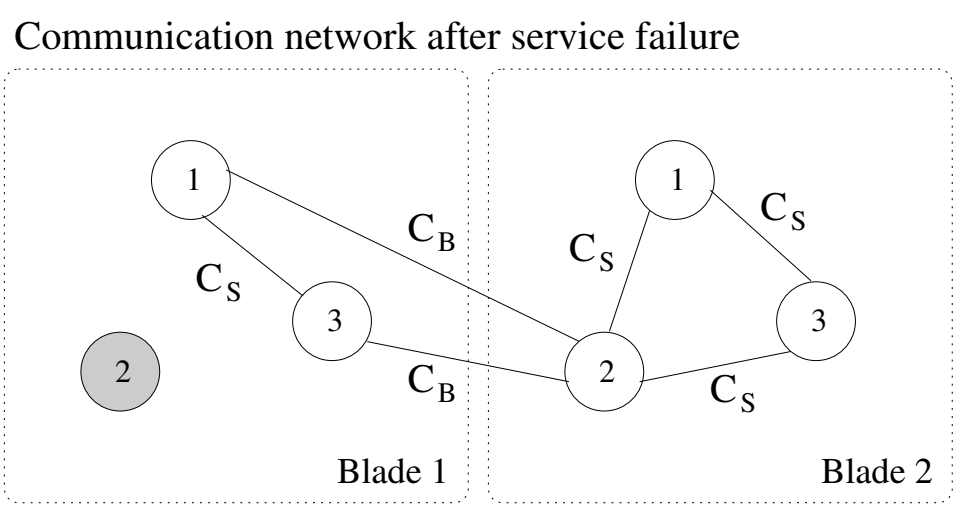
Redundancy Scheduling



Example: 2 jobs, each with 3 parallel tasks, using redundancy 2. Total tasks = 2 * 3 * 2 = 12.

Communication Networks





Tasks communicate with the nearest copy of every other task. Here, all communication is intra-server (on the same physical hardware).

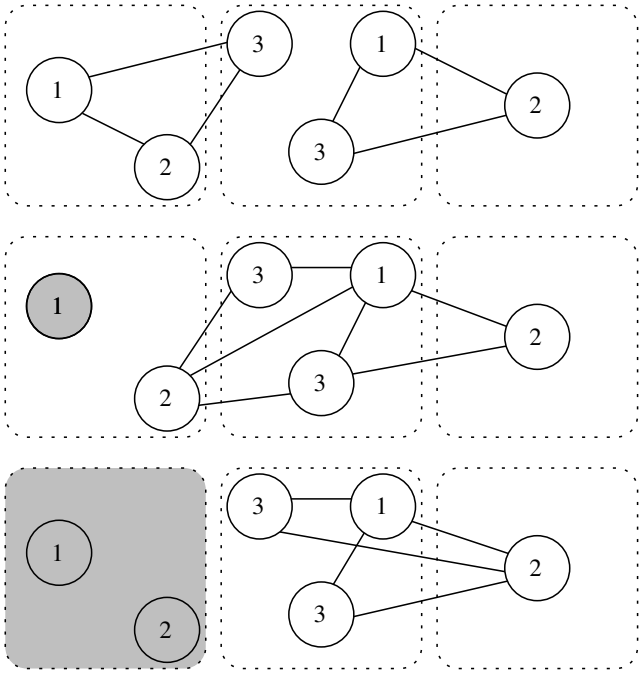
When task 2 fails, communication costs increase from $6C_S$ to $4C_s$ $+2C_B$. We now have more expensive inter-server communication.

Random Scheduling

Initialisation

Service Fail

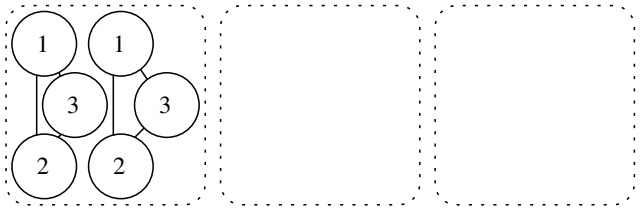
Blade Fail



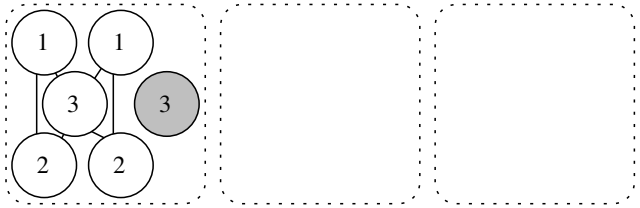
Lots of long-distance (costly) communications, but some resilience to localised failure

Pack Scheduling

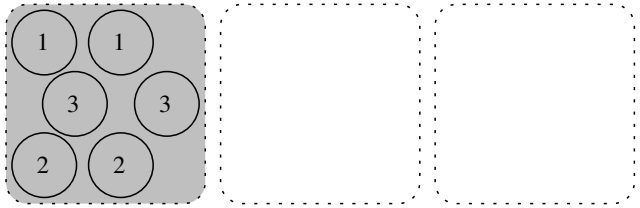
Initialisation



Service Fail



Blade Fail



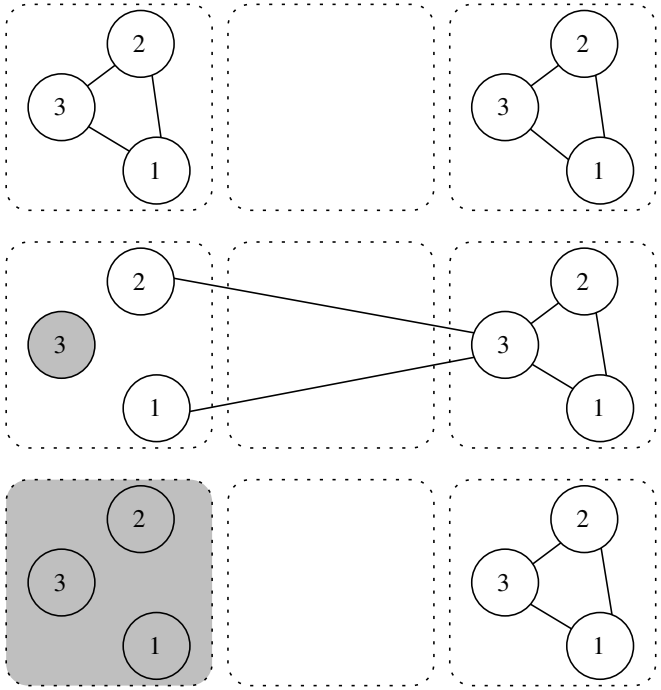
Localised (cheap) communication, but vulnerable to localised hardware failure

Cluster Scheduling

Initialisation

Service Fail

Blade Fail



Mainly localised (cheap) communication, with some resilience to localised failure

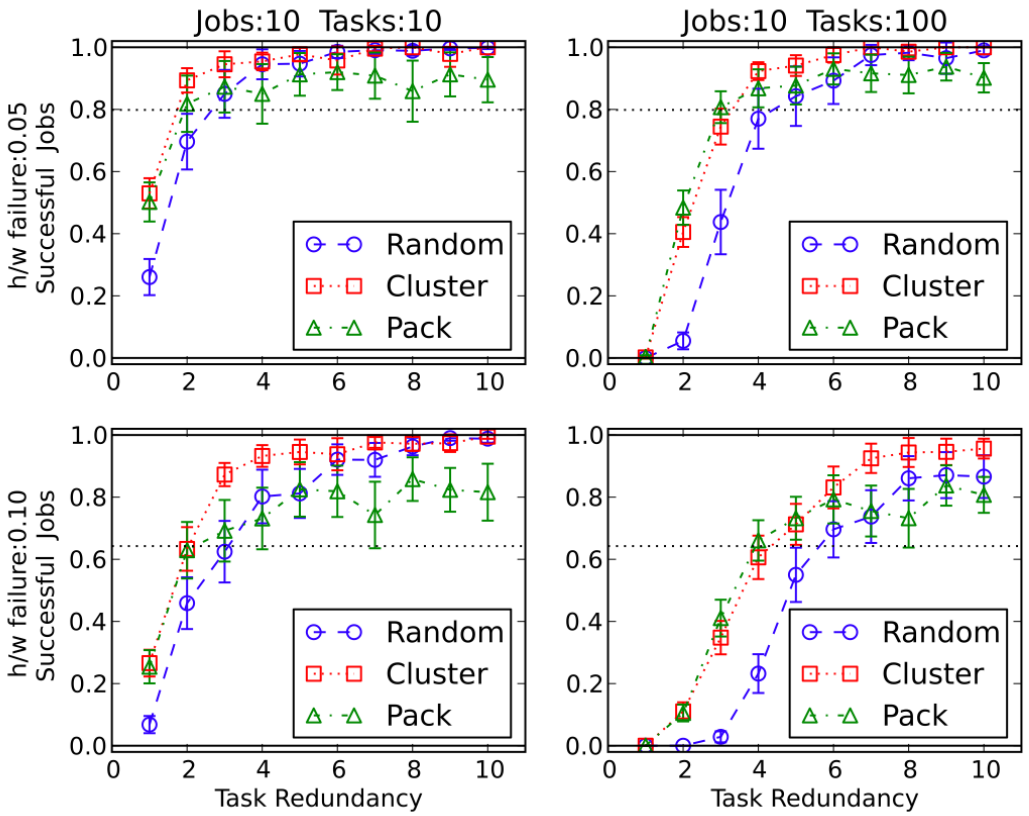
Definition: Job Failure

$$J = \begin{pmatrix} j_{1,1} & j_{1,2} & \cdots & j_{1,r} & \cdots & j_{1,R} \\ j_{2,1} & j_{2,2} & \cdots & j_{2,r} & \cdots & j_{2,R} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ j_{t,1} & j_{t,2} & \cdots & j_{t,r} & \cdots & j_{t,R} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ j_{T,1} & j_{T,2} & \cdots & j_{T,r} & \cdots & j_{T,R} \end{pmatrix}$$

$$fails(J) \Leftrightarrow \exists t \in T, \left\{ \forall r \in R, fails\left(j_{t,r} \right) \right\}$$

A job fails if all redundant copies of a task fail

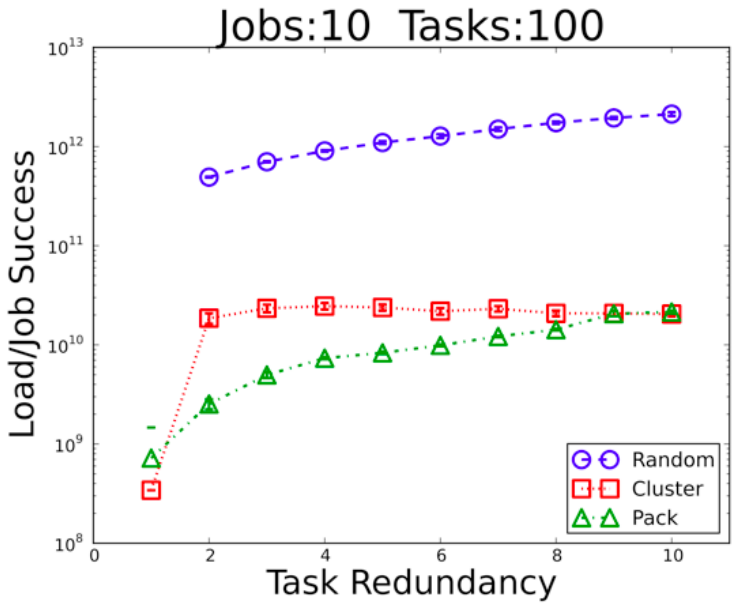
Results: Resilience



20,000 servers

- Increase in failure rate -> decrease in resilience
- Increase in tasks per job -> decrease in resilience
- Overall, Cluster is the most resilient algorithm

Communication Costs of Redundancy



- Random & Pack: costs scale linearly with R
- Cluster: communication costs invariant (R>1)
- Pack most efficient, Random least efficient

Conclusion & Future Work

- Demonstration of how simulation models of cloud-scale DCs can be used to determine the resilience and cost of redundancy scheduling algorithms
- Models will enable cloud providers to tune their schedulers based on expected failure and demand
- Future work: To create an open-source, extensible simulation framework for modeling cloud DCs
  - CloudSim: to be open-sourced in 2012
  - Will include Resilience, Pricing, Power Efficiency, Cooling, etc.

Questions?

Contact:

Dr John Cartlidge

john.cartlidge@bristol.ac.uk

http://www.cs.bris.ac.uk/~cszjpc/

Collaborators:

Ilango Sriram, Prof. Dave Cliff

LSCITS: http://lscits.cs.bris.ac.uk/