Robust Distributed Services in Embedded Networks

Michael Reiter

Take-Away Message

An analogy

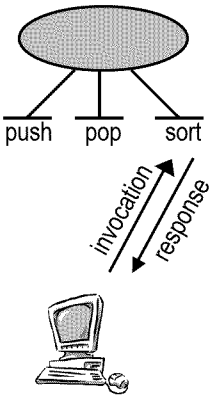
- Users on the Internet are not satisfied with only connectivity
  - Higher-level services attract users and applications
- Same theme is arising in mobile handheld applications
- Similarly, we believe that ensuring connectivity is only part of the picture for embedded / ad-hoc / … networks
- Users and applications will require services, databases, and other "pull-style" information backplanes

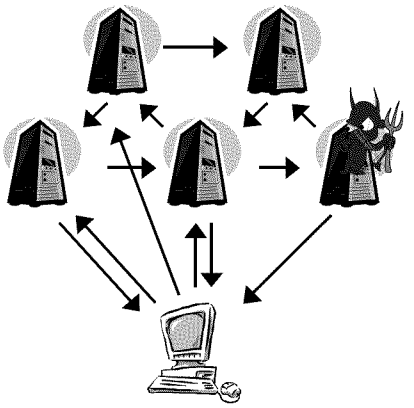
What Makes This Difficult?

- **If your embedded / ad-hoc network is autonomous, it may have no servers!**
  - At least not in the typical sense of that word
- **A server is typically**
  - Well provisioned and maintained
  - Reliably connected
  - Relatively trustworthy
- **Embedded / ad hoc networks may lack any such nodes**

Survivable Distributed Services

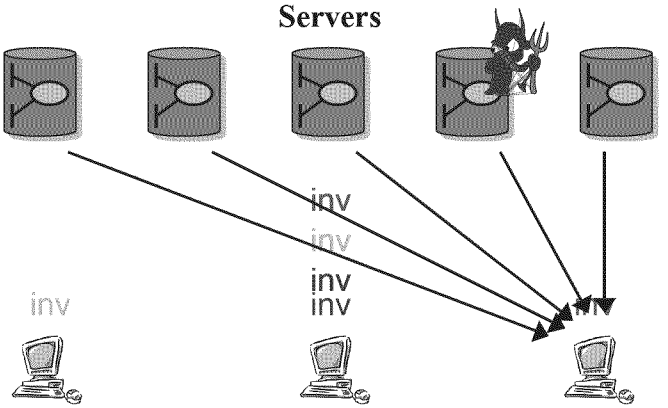
- **Service, or object, abstraction**
- **Implementation**





Traditional Approach: State Machine Replication

5



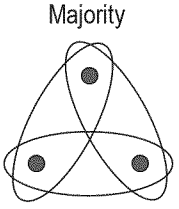
- **Offers no load dispersion, and degrades as system scales**

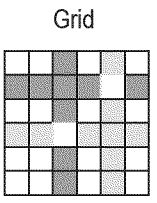
Carnegie Mellon

Quorum Systems

6

- **Quorum systems:**
  - Basic tool for synchronization in distributed systems
  - A set of subsets (*quorums*) of a universe U of logical elements, having intersection property (any pair of quorums intersect)





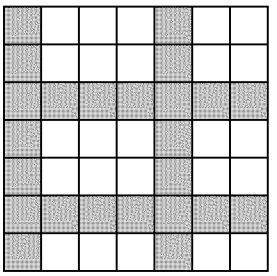
Carnegie Mellon

Byzantine Quorum Systems

- **A quorum system is a data redundancy technique that supports load dispersion among servers**
- **Only a subset of servers are accessed in each operation**
  - Good servers in intersection must be enough to "out vote" bad servers

Construction	Resilience	Quorum size
Threshold		$3n/4$
M-Grid	$b \leq \sqrt{n}/2$	
BoostFPP		
Probabilistic		$O(\max\{b, \sqrt{n}\})$ $O(\sqrt{bn})$

Ex: Grid with n=49, b=3



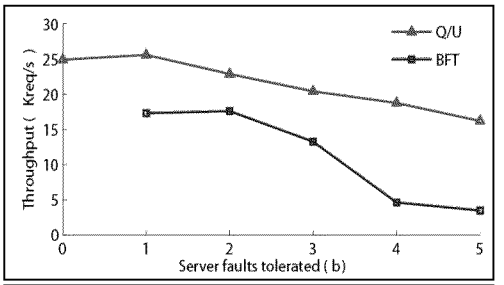
Carnegie Mellon

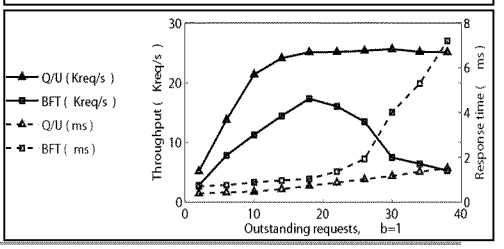
Protocols for Survivable Services

[w/ Abd-El-Malek, Ganger, Goodson, and Wylie]

- **New protocols for**
  - Read/write objects
  - Arbitrary services (Q/U)

  combining
  - Quorum systems
  - Optimistic execution
  - Fast cryptographic primitives
- **Graphs on right show that quorum protocols can scale better than SMR in real systems**
  - But these were well-connected settings





Carnegie Mellon

Dealing with Network Effects

- **Network effects are likely to be just as important in embedded / ad hoc networks as load dispersion**
- **Even worse, minimizing network delays for accessing quorums can be in conflict with load dispersion**
  - May have to bypass a close but heavily-loaded quorum in favor of a less-loaded but more distant quorum
- **Can we balance this tradeoff?**

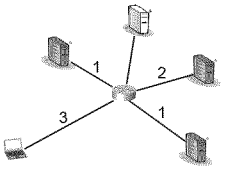
Quorum Placement Problems

- **Place "good" quorum systems on network**
  - to minimize network-specific measures
  - preserve "goodness"
- **Goodness = load**
  - Assume each quorum Q is accessed with probability $p(Q)$
  - $load_p(u) = \Sigma_{Q:\, u \in Q}\, p(Q)$
- **Network measures:**
  - Average delay observed by clients when accessing quorum system
  - Network congestion induced by clients accessing quorum system

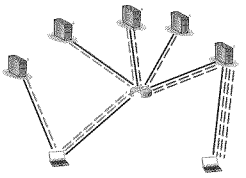
Network Measures

- **Given**
  - network $G = (V, E)$
  - delay $d : E \rightarrow \mathsf{R}^+$
  - edge_cap: $E \rightarrow \mathsf{R}^+$
  - quorum system $\mathcal{Q}$ over U
  - access strategy p: $\mathcal{Q} \rightarrow [0, 1]$
  - placement $f : U \rightarrow V$

- **Average max-delay:**
  - $d(v, f(Q)) = \max_{u \in Q} d(v, f(u))$
  - $d(v, f(\mathcal{Q})) = \mathsf{E}_p[d(v, f(Q))] = \Delta_f(v)$
  - $\text{avg\_delay}_f = \text{Avg}_{v \in V}[\Delta_f(v)]$

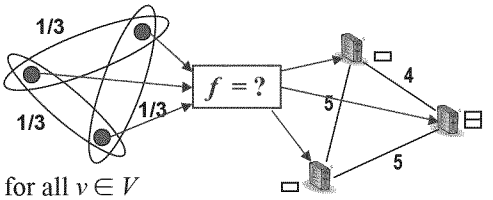


- **Network congestion:**
  - flow $g_{v,f(u)}$: $E \rightarrow \mathsf{R}^+$
  - $\text{traff}_e(v, f(Q)) = \Sigma_{u \in Q} g_{v,f(u)}(e)$
  - $\text{traff}_e = \text{Avg}_{v \in V} \mathsf{E}_p[\text{traffic}_e(v, f(Q))]$
  - $\text{cong}_f = \max_{e \in E} \text{traff}_e / \text{edge\_cap}(e)$



Carnegie Mellon

Quorum Placement Problem for Delay (QPPD)



- **Given**
  - graph $G = (V, E)$,
    - with distances d: $E \rightarrow \mathsf{R}^+$
    - and capacity *node_cap(v)* for all $v \in V$
  - a quorum system $\mathcal{Q}$
    - with a distribution p s.t. each Q_i is accessed with prob. $p(Q_i)$

- **find placement f**
  - minimizing average max-delay, $\text{Avg}_{v \in V}[\Delta_f(v)]$
  - subject to load constraints: $load_f(v) \leq node\_cap(v)$, for all $v \in V$

Carnegie Mellon

Results for QPPD

[w/ Gupta, Maggs, Oprea]

- **QPPD is NP-hard**
- **For any $\alpha > 1$, there is a $(5\alpha/(\alpha-1), \alpha+1)$ approximation:**
  - If we allow capacities to be exceeded by a factor of $\alpha+1$, then we can achieve average max-delay within a factor of $5\alpha/(\alpha-1)$ of optimal for all capacity-respecting solutions
- **For Majority and Grid, if node capacities equal the optimal load of the quorum system, there is a $(5, 1)$-approximation.**

Quorum Placement for Congestion (QPPC)

- **Two routing models:**
  - Fixed paths (given as input)
  - Arbitrary paths (chosen probabilistically)
- **Given:**
  - graph $G = (V, E)$,
    - node capacities *node_cap(v)* for all $v \in V$,
    - and edge capacities *edge_cap(e)* for all $e \in E$
  - a quorum system $\mathcal{Q}$
    - with a distribution p s.t. each Q_i is accessed with prob. $p(Q_i)$
- **find placement f**
  - minimizing max relative-congestion, $\text{Max}_{e \in E}[\text{cong}_f(e)]$
  - subject to load constraints: $load_f(v) \leq node\_cap(v)$, for all $v \in V$

Results for QPPC

[w/ Golovin, Gupta, Maggs, Oprea]

- **QPPC is NP-hard in either model**
  - Even finding any node-capacity-respecting solution is NP-hard
- **Arbitrary paths:**
  There is an $(O(\log^2 n \log\log n), 2)$-approximation.
  - If we allow node capacities to be exceeded by a factor of **2**, then we can achieve max relative-congestion to within a factor of $O(\log^2 n \log\log n)$ of optimal for all node-capacity-respecting solutions

  If G is a tree, there is a $(5, 2)$-approximation.
- **Fixed paths:**
  There is an $(O(\eta \log n / \log\log n), 2)$ –approximation, where η is the size of the set $\{ \lfloor \log_2(\text{load}(u)) \rfloor \mid u \in U \}$

Theory vs. Practice

- **We have some initial theory results**
  - But many theoretical questions remain unanswered
- **But how does the theory correspond to practice?**
  - Example: Network delay is only one component of client response time, the other being server load
  - So, network delay and server load are not easily separable for this measure
- **These problems still need to be explored even in fixed-infrastructure networks**

Embedded / Ad Hoc Networks

- **Importance of addressing faults**
  - Not only due to disabling quorum elements, but also due to impinging on quorum reachability
- **If population is dynamic**
  - Need to consider migrating quorum elements
- **If mobility is involved**
  - Continually need to re-evaluate quorum placements