Analysis of P2P Storage Systems

Réunion du Boréon

March 13, 2009

Program

- Simulating millions of nodes and billions of events: OSA - BROCCOLI - SPREADS.
  Participants: **Judicael**, Olivier.
- P2P storage systems and data placement.
  Participants: fred, **Julian**, Stéphane.
- Models of P2P storage systems under resource constraints.
  Participants: **fred**, Julian, Stéphane.
- Some future directions.
  Participants: **fred**, Judicael, Julian, Olivier, Stéphane.

Models of P2P storage systems under resource constraints.

Participants: fred, Julian, Stéphane.

Problem: Motivation and Related Work

- Few theoretical models for P2P storage systems.
- All models assume unlimited bandwidth.
  - Useful for network provisioning.
  - But far from behavior of real systems if used bandwidth close to available bandwidth.
- $\longrightarrow$ need for models with limited bandwidth.

Difficulties

- Unlimited bandwidth $\longrightarrow$ block reconstructions independent.
- Limited bandwidth $\longrightarrow$ strong dependencies (the bandwidth is shared).
- $\longrightarrow$ the reconstruction times become longer.
- $\longrightarrow$ more data losses.

Network models and bandwidth limits

- Typically, network: graph, bandwidth limits: capacity on each edge.
- Model for our application:
  - Limiting bandwidth: access (peer) bandwidth.
  - The connecting backbone network: unlimited capacity.
- Remark: Different from a model where the global (sum) bandwidth is limited.
- Remark: homogeneus peers or not (peers with different capacities).

Model of the access link

- Asymetric link (DSL): typically download 2-10 times larger than upload.
- Models:
  1. For each peer: BW_d and BW_u
  2. Simplification (valid in case of strong asymmetry):
     - limited upload bandwidth
     - unlimited download bandwidth

  - Discussion: condition of validity of this simplification.

Modeling of the disk loss event

- Blocks to be reconstructed: state $r(b) \geq r_0 + 1 \rightarrow r(b) \leq r_0$.
- Problem: what is the number of such blocks for each disk loss?
- Depends on two factors:
  - Number of fragments in the disk.
    Models:
    - simple model: same for all disk at each time step $\approx \frac{B(s+r)}{N}$
    - refined model: geometric
  - Proportion of fragments in state $r_0 + 1$.
    Model:
    - same for all disks and at each time step (of course after a mixing time)
    - around $\frac{1}{r-r_0}$ [figure]

Modeling of the disk loss event

- Conclusion: at each disk loss, β blocks go into reconstruction with

$$\beta \approx \frac{B(s+r)}{N(r-r_0)}$$

- With refined model:

$$\beta \approx \frac{B(s+r)}{N(r-r_0)} G(1),$$

with $G(1)$, a truncated normalized geometric distribution.

Goal

- We want to obtain:
  - Needed (upload) bandwidth (# of reconstructions).
  - System data losses.
- Data losses directly related to the reconstruction time: the greater the reconstruction time, the larger the probability to die.
- More precisely, loss of $\geq r_0$ (redundancy at the start of the reconstruction) during the reconstruction $\Rightarrow$ block dies.
- If the reconstruction lasts θ:

$$\Pr[die|W=\theta] = \binom{s+r_0}{r_0}(1-(1-\alpha)^\theta)^{r_0}((1-\alpha)^\theta)^s$$

with α probability for a disk to die during a time step.

Goal

- Hence the probability to die during a reconstruction, P_D, with

$$P_D = \sum_{i=0}^{\infty} \Pr[\text{die} | W = i] \Pr[W = i].$$

- The number of dead blocks during a time T, D_T, is then obtained by the number of reconstruction during T, R_T, by

$$D_T = P_D R_T.$$

- $\rightarrow$ Our interest: distribution of the reconstruction time.

Goal

- Remark: distribution and not only the expectation. [figure] distribution.
- Dead blocks can come from:
  - the few blocks with long reconstruction time,
  - the majority of blocks that have an average reconstruction time,
- Main cause will depend on system parameters.

A block reconstruction: 4 phases

- Discovery.
- Retrieval: download s fragments from s peers. [figure]
- Reconstruction: Matrice inversion.
- Sending: send $r - r_0$ missing fragments. [figure]

Computing the system throughput: A matching problem

- System throughput: find a maximal BW_u-matching in a bipartite graphe $G = (V_1 \cap V_2, E)$. [figure]
- V_1 : blocks, V_2 : peers, Edges: (i,j) Peer j has to send a fragment of block i.
- Metric of efficiency: ratio $\rho = \frac{|M|}{N.BW_u}$.
- Questions:
  - How to determine this ratio in function of the system parameters?
  - In which case is this ratio 1 (maximal efficiency)?

A matching problem

- Goal: plug this ratio and put it as parameter in the model.
- Intuition: ratio depends on the edge density (load of the system).
- Lots of issues:
  - Centralized computation: polynomial
  - Distributed computation: ?? block reconstruction scheduling.
  - What is the performance of a greedy algo: take first the nodes with few edges (most advanced reconstruction) (avoid starvation).

A first model with queues

- A peer has a queue: with the number of blocks to reconstruct. [figure]
- Disk failure with proba α: β/N blocks go in the queue.
- Reconstruction: at each top a peer handles k block reconstuctions with $k = \frac{BW_u}{r - r_0 + 1}$
  - 1 for retrieval
  - $r - r_0$ for sending

  Normalization.

Validity of the problem

- Lots of simplifications:
  - disk crash: β/N for each node
    - same number of fragments for each disk at each time
    - same fraction of blocks in state $r_0 + 1$
    - block reconstruction well distributed among peers
  - the processing is done "at full speed"
    - ratio —matching—/bandwidth=1
    - the retrieval phase is done in 1 time step
- Questions:
  1. Can we analyze this model?
  2. Is the model close to the real system? For which set of parameters?
  3. Which refinements can/have to be introduced?

Analysis of the queuing model

- Model: $M/D/1/\infty$ queing model with batched/bulk arrivals of constant size:
  - Arrivals follow a Poisson process.
  - Deterministic service.
  - 1 server.
  - Queue of infinite size.
- It is not one of the classical models. Some papers on MDc with batches of exponentiel size.
- Write the queue generating function. Computations:
  - Get the asymptotic of the coefficients (Proba to have a large queue is exponentially low)
  - Compute numerically the first coefficients.
- Give the service time and so the reconstuction time.

Is the model close to the real system?

First set of simulations:

- Ratio —matching—/BW: [Figure]
- Reconstruction times: [Figure]
- Block losses: [Figure]

Possible things to do

Model refinements:

- Geometric disk sizes.
- Impact of the size of the queue: a filled disk has to send more fragments during the retrieval phase.
- Ratio —matching—/BW.

Things to do:

- analyze local placement: harder as some nodes have a lot more reconstruction to do, so are naturally system bottlenecks.

Future directions

- Comparison of different reconstruction policies.
- Multiple failures.
- P2P streaming systems.

Comparisons of different reconstruction policies

- Which blocks have to be reconstructed?
  Best policy: saddle, eager, probabilistic saddle, . . .
- In which order?
  Scheduling (blocks with less redundancy, blocks with most advanced reconstruction, . . .)
- By who?
  Biased Reconstruction policies: e.g. disks with a large number of blocks should be in charge of less reconstructions.
  Shuffling policies.

Multiple failures

Problems:

- System growth.
- Catastrophe analysis (multiple failures).
- Attack (server flooding, . . .).

Model: Introduction of rare events in the model.

P2P Streaming Systems

- Neighboring problem studied inside ANR Aladdin.
  Participants: INRIA GANG (Viennot), LIAFA (de Montgolfier). And also: Orange labs (Mathieu), Thomson (Massoulié).
- Problem: Diffusion of live streaming through P2P overlays.
- Main applications: live soccer games.
- Existing systems: CoolStreaming, PPLive, SopCast, Tvants.

- Use random epidemic-style not structured diffusion schemes: (stream divided into small chunks that follow random, independent paths in the peer population)
- $\neq$ structured systems that builds a multicast overlay by means of one or several static spanning trees.
- Scalable and Robust: Particularly suitable for Internet (dynamic, heterogeneus).
- Question: Analysis of the P2P Streaming Systems.

Model

Epidemic Live Streaming: Optimal Performance Trade-Offs
Thomas Bonald, Laurent Massoulié, Fabien Mathieu, Diego Perino, Andrew Twigg.

- One source and N peers.
- Source:
  - creates sequence of chunks numbered 1,2,3,..., at rate λ.
  - sends each chunk to one peer, chosen uniformly at random.
- Dissemination to the N peers achieved by the peers ($s(u)$ upload speed of node u).
- Peers have a partial knowledge of the nodes: Directed graph $G = (V, E)$ and $(u, v) \in E$ if u knows v (u can send a chunk to v).
- $C(u)$ collection of chunks u has received.

Diffusion schemes

Dissemination to the N peers. Schemes are combinations of

- push-based/pull-based: transmission initiated by sender/receiver
- Choice of a peer:
  - random peer:
  - random useful peer:
  - most deprived peer:
- Choice of a chunk:
  - last blind chunk:
  - latest useful chunk:
  - most recent useful chunk:
  - random useful chunk:

Model hypothesis:

- Discrete time.
- Source sends $\lfloor \lambda \rfloor$ chunk per time slot $+$ one with proba $\lambda - \lfloor \lambda \rfloor$.
- Perfect/Imperfect knowledge: intended transmission to our neighbors are known.

Diffusion rate/Diffusion delay

Performance metrics:

- Diffusion function r, $r(t)$ probability that it takes no more than t time slots for arbitrary chunk to arbitrary peer. [figure]
- Diffusion rate: asymptotic $r(t)$ when $t \to \infty$.
- Diffusion delay: time to be at $(1 - \varepsilon)$ diffusion rate.

Results

- *random peer, latest useful chunk mechanism* can achieve dissemination at an optimal rate and within an optimal delay, up to an additive constant term.
- $\longrightarrow$ epidemic live streaming algorithms can achieve near-unbeatable rates and delays.
- recursive formulas for the diffusion function of two schemes referred to as *latest blind chunk, random peer* and *latest blind chunk, random useful peer.*

Things to do

- For us: Bibliography.
- Building and evolution of the overlay graph.
- Frequency and size of control messages.
- Robustness to cheating and selfish behavior.