A Simulation-based Evaluation of a Hybrid Storage System combining P2P, F2F, and Cloud storage with a Distributed Reputation System

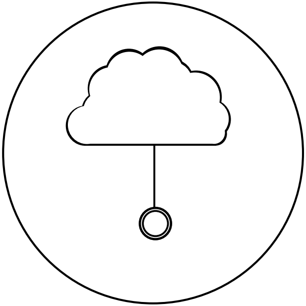
Anders Skoglund

andsk668@student.liu.se

November 04, 2013

Storage methods

- Cloud storage
- P2P storage
- F2F storage
- Hybrid storage + reputation system

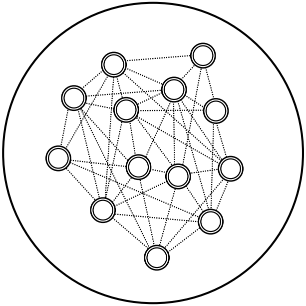


Advantages

+ Scalable
+ High availability
+ Contractual accountability

Disadvantages

− Cost
− Possible single point of failure

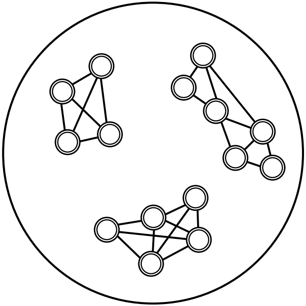


Advantages

- + Large number of peers
- + Scalable
- + No single point of failure
- + Cooperative / low cost

Disadvantages

- – Semi-anonymous peers
- – No accountability
- – Peers can't be trusted

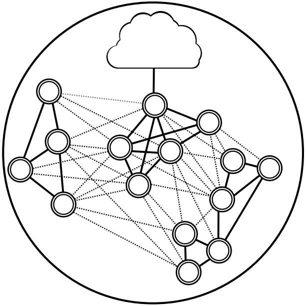


Advantages

- \+ No single point of failure
- \+ Cooperative / low cost
- \+ Social accountability
- \+ Known and trusted peers

Disadvantages

- – Few peers
- – Possible availability issues



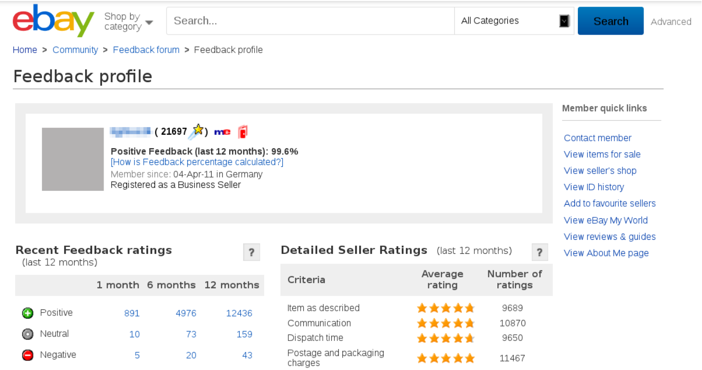
Advantages

+ Scalable
+ Trusted friend peers
+ Predictable peer behavior
+ No single point of failure
+ Cooperative / low cost

Disadvantages

– ?

- DHT
- Distributed search
- Block distributor
- Reputation system



ebay Shop by category | Search... | All Categories | Search | Advanced

Home > Community > Feedback forum > Feedback profile

Feedback profile

(21697) me

Positive Feedback (last 12 months): 99.6%
[How is Feedback percentage calculated?]
Member since: 04-Apr-11 in Germany
Registered as a Business Seller

Member quick links

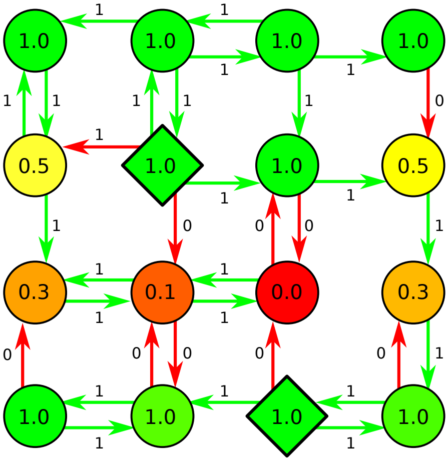
- Contact member
- View items for sale
- View seller's shop
- View ID history
- Add to favourite sellers
- View eBay My World
- View reviews & guides
- View About Me page

Recent Feedback ratings (last 12 months) ?

	1 month	6 months	12 months
Positive	891	4976	12436
Neutral	10	73	159
Negative	5	20	43

Detailed Seller Ratings (last 12 months) ?

Criteria	Average rating	Number of ratings
Item as described	★★★★★	9689
Communication	★★★★★	10870
Dispatch time	★★★★★	9650
Postage and packaging charges	★★★★★	11467


1.0 1.0 1.0 1.0
0.5 1.0 1.0 0.5
0.3 0.1 0.0 0.3
1.0 1.0 1.0 1.0

Algorithm 1 BasicEigenTrust

C	:	Local trust matrix.
$\vec{p}$	:	Vector of relative trust values for all pre-trusted peers.
$\vec{t}$	:	Vector of global trust values for all peers.
a	:	Weight given to pre-trusted peers when computing global trust.
ϵ	:	Threshold used to stop the algorithm once it converges.

function COMPUTETRUST(C)
    $\vec{t}^{\,0} \leftarrow \vec{p}$
    $k \leftarrow 0$
    repeat
        $\vec{t}^{\,k+1} \leftarrow (1-a)C^T\vec{t}^{\,k} + a\vec{p}$
        $\delta \leftarrow ||\vec{t}^{\,k+1} - \vec{t}^{\,k}||$
        $k \leftarrow k+1$
    until $\delta < \epsilon$
    return $\vec{t}^{\,k}$
end function

Reputation systems
Distributed EigenTrust

Algorithm 2 SecureEigenTrust

C	:	Local trust matrix.
$\vec{p}$	:	Vector of relative trust values for all pre-trusted peers.
$\vec{t}$	:	Vector of global trust values for all peers.
A_d	:	Peers that have reported local trust values about a daughter peer d.
B_d	:	Peers that a daughter peer d has reported local trust values about.
D	:	Daughter peers of the score manager.
M_i	:	All score managers for the peer i.
a	:	Weight given to pre-trusted peers when computing global trust.
ϵ	:	Threshold used to stop the algorithm once it converges.

function ComputeTrust(C)
  for each $d \in D$ **do**
    $A_d \leftarrow$ ServedByDaughter(d)
    $B_d \leftarrow$ HasServedDaughter(d)
    $k \leftarrow 0$
    for each $j \in A_d$ **do**
      $c_{jd} \leftarrow$ QueryLocalTrust(Hash(j))
    end for
    repeat
      $t_d^{k+1} \leftarrow (1-a)\sum_{j=1}^{n} c_{jd} t_j^k + a p_d$
      for each $j \in B_d$ **do**
        $M_j \leftarrow$ Hash(j)
        SendLocalTrust(c_{dj}, M_j)
        SendGlobalTrust(t_d^{k+1}, M_j)
      end for
      for each $j \in A_d$ **do**
        $M_j \leftarrow$ Hash(j)
        $c_{jd} \leftarrow$ RecieveLocalTrust(M_j)
        $t_j^{k+1} \leftarrow$ RecieveGlobalTrust(M_j)
      end for
      $k \leftarrow k+1$
    until $|t_d^{k+1} - t_d^k| < \epsilon$
  end for
end function

$$s_{ij} = \mathrm{sat}(i,j) - \mathrm{unsat}(i,j)$$

$$c_{ij} = \begin{cases} \frac{\max(s_{ij},0)}{\sum_j \max(s_{ij},0)}, & \text{if } \sum_j \max(s_{ij}) \neq 0 \\ p_j, & \text{otherwise} \end{cases}$$

$$p_i = \begin{cases} \frac{1}{|P|}, & \text{if } i \in P \\ 0, & \text{if } i \notin P \end{cases}$$

$$t_j = (1-a)\sum_i c_{ij} t_i + a p_j$$

- \+ Simple
- \+ Well analyzed
- \+ Scalable
- − Very simple trust model
- − Relative trust values

$$s_{ij} = \begin{cases} \frac{\text{sat}(i,j)}{\text{sat}(i,j)+\text{unsat}(i,j)}, & \text{if } \text{sat}(i,j) + \text{unsat}(i,j) \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

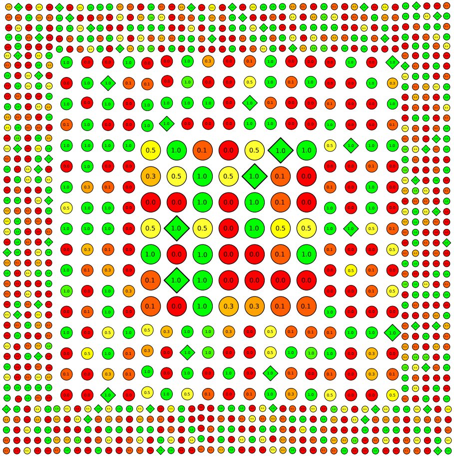
$$c_{ij} = s_{ij}$$

$$t_j = \begin{cases} \sum_i c_{ij} w_{ij}, & \text{if } j \notin P \\ 1, & \text{if } j \in P \end{cases}$$

$$w_{ij} = \frac{t_i}{\sum_{k \in A_j} t_k}$$

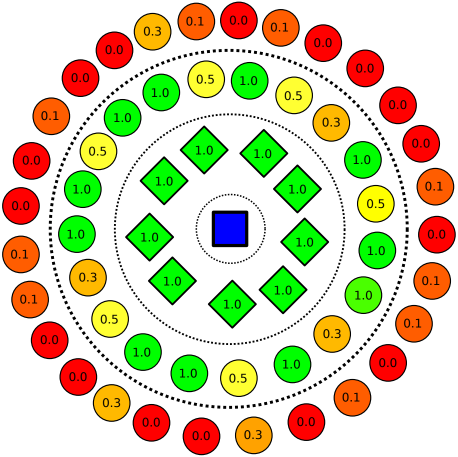
- Computes more useful trust values than EigenTrust
- Can use the same distributed algorithms as EigenTrust

Reputation systems
Global trust



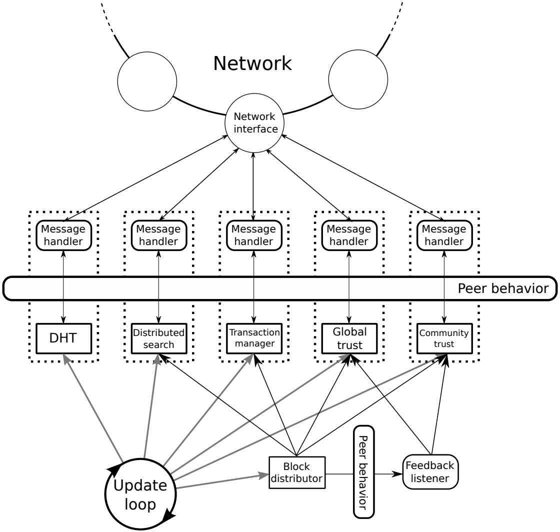
Global trust (P2P)

- Single trust value per peer
- Scalable
- Many malicious peers



Community trust (F2F)

- Separate trust values are computed by each peer
- Not scalable
- More trusted peers
- Fewer peers and resources
- Can get stuck in local maximum



Network
Network interface
Message handler
Message handler
Message handler
Message handler
Message handler
Peer behavior
DHT
Distributed search
Transaction manager
Global trust
Community trust
Update loop
Block distributor
Peer behavior
Feedback listener

Honest peer

- Always fulfill transactions
- Give honest ratings

Malicious peer

- Always fulfill transactions with malicious peers
- Otherwise return “corrupt” data with probability p_m
- Always gives positive ratings to malicious peers
- Otherwise give false (negative) ratings

Distribution policies

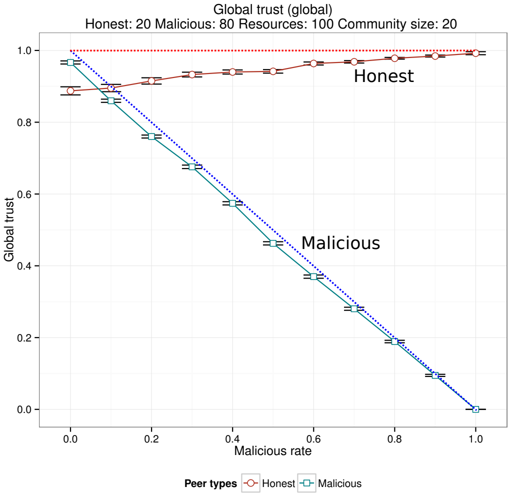
- *Global* : Only use global (P2P) peers
- *Community* : Only use community (F2F) peers
- *Mixed* : Use both global (P2P) and community (F2F) peers

File types

- *0.0* : Trust $\geq$ 0.0
- *0.3* : Trust $\geq$ 0.3
- *0.6* : Trust $\geq$ 0.6
- *0.9* : Trust $\geq$ 0.9

Results

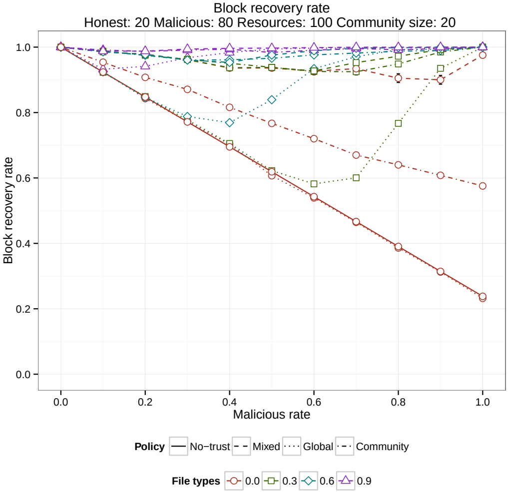
Malicious rate × Average global trust



- Ideal would be:
  $T_{honest} \approx 1$
  and
  $T_{malicious} \approx$ malicious rate
- Global trust is close to the ideal value for both honest and malicious peers

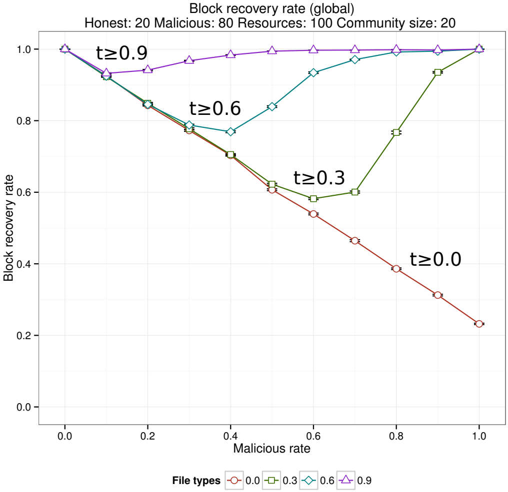
Results

Malicious rate × Block recovery rate

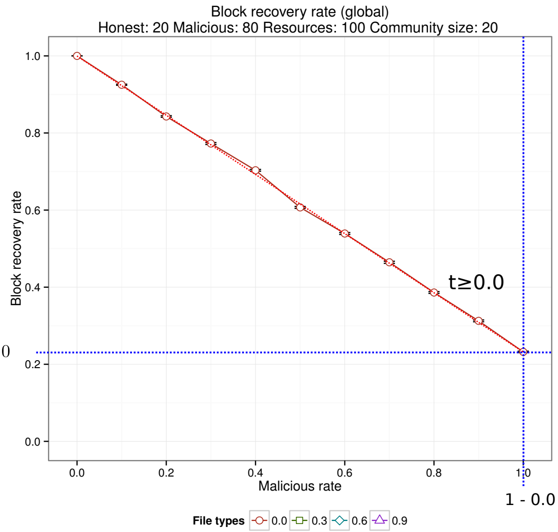


Results

Malicious rate × Block recovery rate (Global peers)



Results
Malicious rate × Block recovery rate (Global peers)

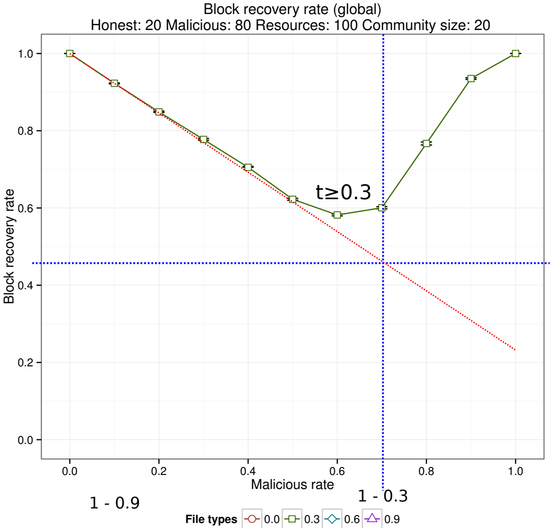


$\frac{20+5}{105} + \frac{80}{105} * 0.0$

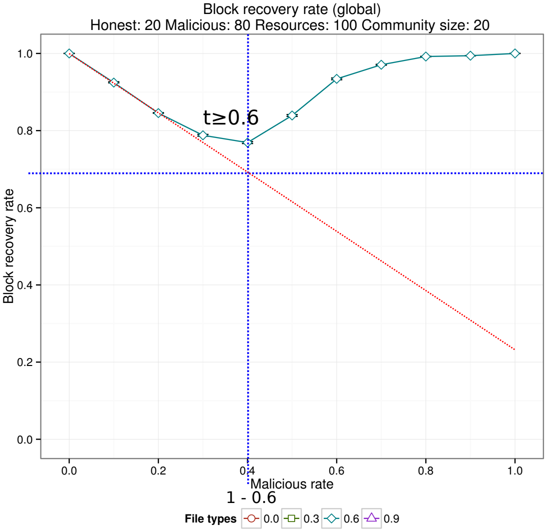
Results

Malicious rate × Block recovery rate (Global peers)

$$\frac{20+5}{105} + \frac{80}{105} * 0.3$$

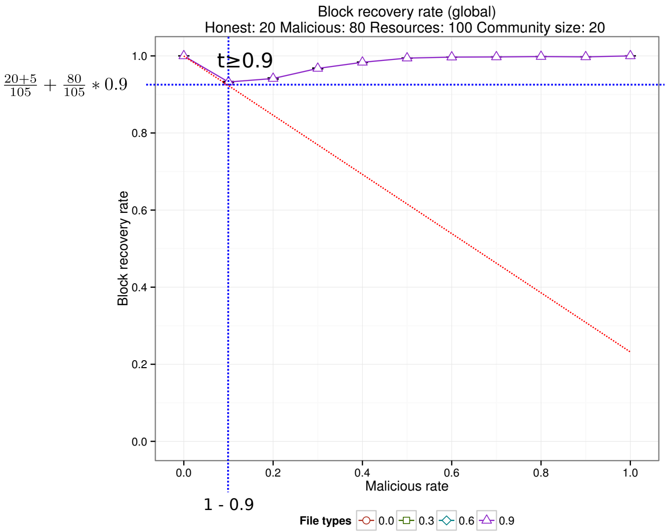


$\frac{20+5}{105} + \frac{80}{105} * 0.6$



Results

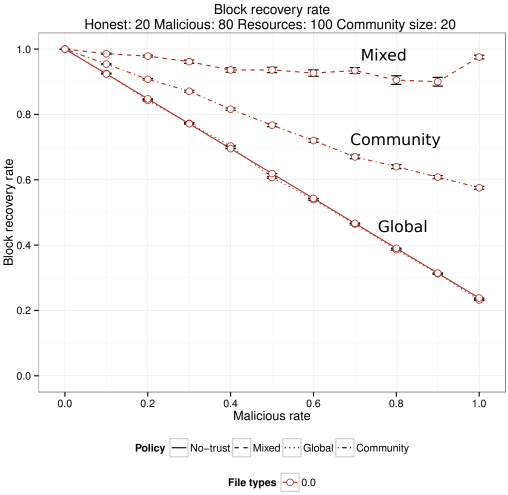
Malicious rate × Block recovery rate (Global peers)



Results

Malicious rate × Block recovery rate (Trust ≥ 0.0)

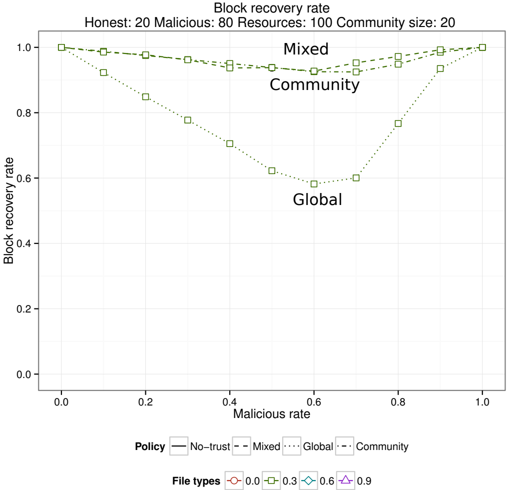
Mixed > Community > Global



Results

Malicious rate $\times$ Block recovery rate (Trust ≥ 0.3)

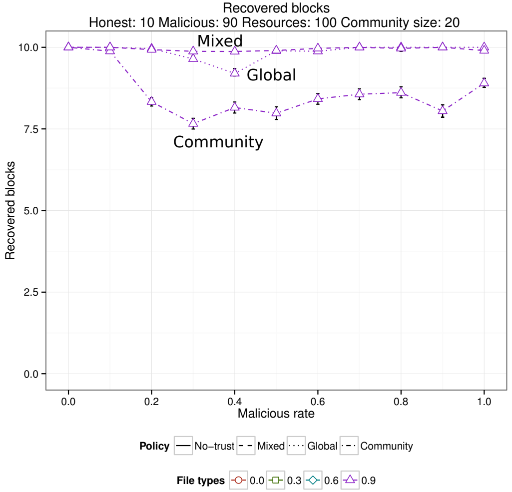
Mixed > Community > Global



Results

Malicious rate $\times$ Block recovery rate (Trust $\geq$ 0.9)

Mixed > Global > Community



Results

- Global trust (P2P) performs better when you need a large number of peers. It is possible to compensate for low trust, and it will perform well as long as the user has chosen appropriate trust requirements.
- Community trust (F2F) performs better when there are enough peers and resources available in the community graph. It requires much less care when choosing trust requirements, but can easily fail if there are not enough peers available.
- A combination of global (P2P) and community (F2F) trust performs at least as well as the best of the two, and often better.

Conclusions

- A hybrid system could work, but some improvements must first be made.
- Both the P2P and F2F part of the system performs better than expected.
- Combining P2P and F2F gives the best performance.
- But using a reputation system may be the most important part.

Possible improvements

Create a better model of peer availability and storage costs

Without a better model of availability or storage cost, and a distribution algorithm that can use it, there is no point in simulating cloud storage.

Find a better trust model

EigenTrust has limitations that could make it a bad choice for this task.

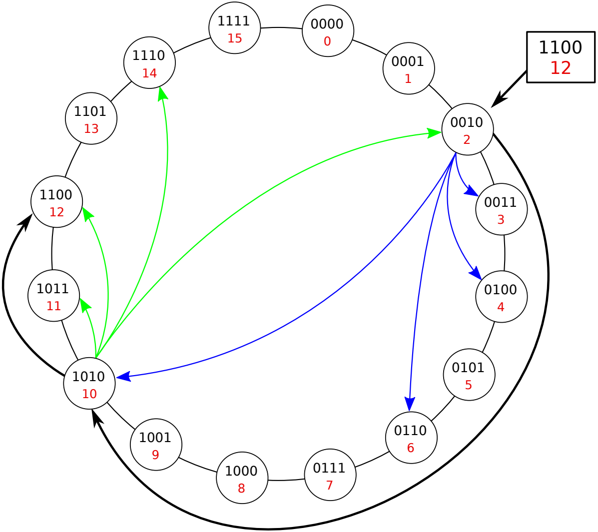
Generalize

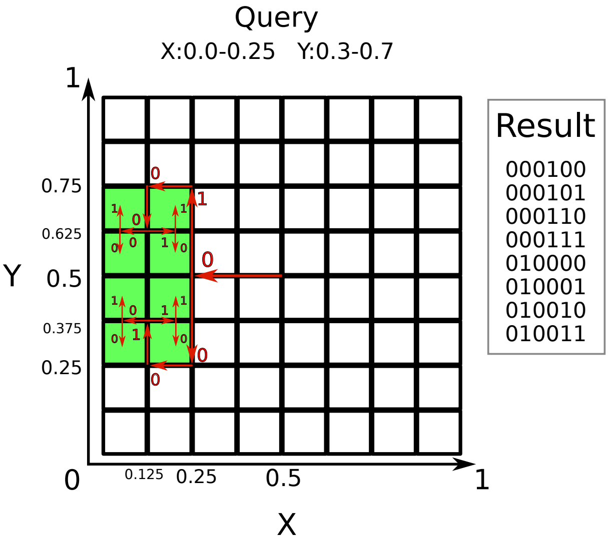
This kind of system should be able to handle any task that can be described as a transaction. It should be possible to use this to create a platform for trading/sharing other types of resources and use it for, among other things, distributed computing.

That's all

Questions?

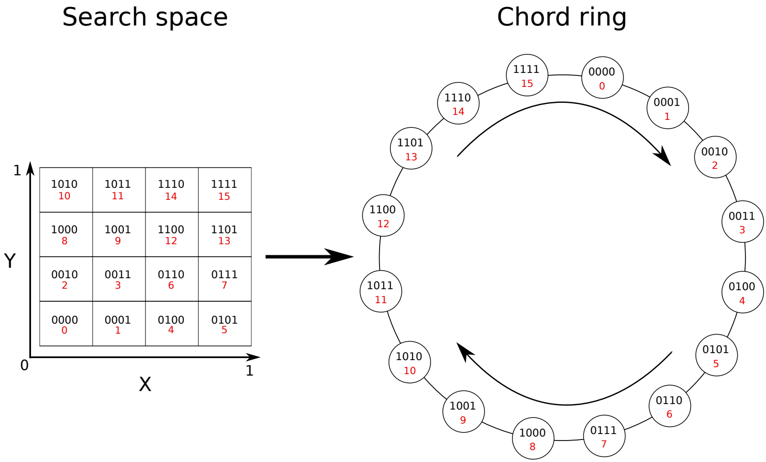
Additional details
DHT





Query
X:0.0-0.25 Y:0.3-0.7
1
0.75
0.625
Y
0.5
0.375
0.25
0
0.125
0.25
0.5
1
X
Result
000100
000101
000110
000111
010000
010001
010010
010011

Additional details
Distributed search



Additional details

Block distributor