Analise de algoritmos distribuídos híbridos para consenso e rede

Fernando Augusto Medeiros Silva

Orientador
Ricardo Augusto de Oliveira Rabelo

Seminários PPGCC, Universidade Federal de Ouro Preto
13 de julho de 2011

Sumário

Motivação

- Aumento do número de dispositivos móveis disponíveis

Motivação

- Aumento do número de dispositivos móveis disponíveis
- Apareciemento de capacidades Ponto a Ponto nesses dispositivos

Motivação

- Aumento do número de dispositivos móveis disponíveis
- Apareciemento de capacidades Ponto a Ponto nesses dispositivos
- Uso de processamento local para atingir objetivo comum

Motivação

- Aumento do número de dispositivos móveis disponíveis
- Apareciemento de capacidades Ponto a Ponto nesses dispositivos
- Uso de processamento local para atingir objetivo comum
- Utilização desde informações turísticas a preferências pessoais

Motivação

- Aumento do número de dispositivos móveis disponíveis
- Apareciemento de capacidades Ponto a Ponto nesses dispositivos
- Uso de processamento local para atingir objetivo comum
- Utilização desde informações turísticas a preferências pessoais

Trabalhos relacionados

- O artigo baseia-se na verficação do modelo de LI [1] para computação de consenso didstribuído utilizando clusters
- Comparaçõo com trabalho de Xiao [2] e Boyd [3].
- Li alega um ganho de $\Omega(logn)$ sobre as abodagens tradicionais

Sistema distribuído

1. Sistema de memória distribuída baseado em troca de mensagens.
2. Representação grafo G(V,E) onde V representa cada tarefa e E representa a comunicação
3. Rede sem fio modelada pelo grafo geográfico randômico $G(n, r(n))$

Definição

1. Considera-se um sistema síncrono

Definição

1. Considera-se um sistema síncrono
2. Rede com n processos

Definição

1. Considera-se um sistema síncrono
2. Rede com n processos
3. Cada processo inicia com um valor V_0

Definição

1. Considera-se um sistema síncrono
2. Rede com n processos
3. Cada processo inicia com um valor V_0
4. A rede interage por troca de mensagens

Definição

1. Considera-se um sistema síncrono
2. Rede com n processos
3. Cada processo inicia com um valor V_0
4. A rede interage por troca de mensagens
5. Ao final cada processo possui o mesmo V final

Técnicas de Busca de Consenso em rede

- Centralizadas
  - Provocam gargalo
  - Um ponto central de falhas
- Distribuídas

Técnicas de Busca de Consenso em rede

- Centralizadas
  - Provocam gargalo
  - Um ponto central de falhas
- Distribuídas
  - Troca de mensagens
  - solução converge para o resultado
  - Utiliza o processamento de cada nó

Técnicas de Busca de Consenso em rede

- Centralizadas
  - Provocam gargalo
  - Um ponto central de falhas
- Distribuídas
  - Troca de mensagens
  - solução converge para o resultado
  - Utiliza o processamento de cada nó

Técnicas Distribuídas para média analisados por Li 2009

- Flood Simples

Técnicas Distribuídas para média analisados por Li 2009

- Flood Simples
- Random Gossip

Técnicas Distribuídas para média analisados por Li 2009

- Flood Simples
- Random Gossip
- Fixed Linear Interaction

Técnicas Distribuídas para média analisados por Li 2009

- Flood Simples
- Random Gossip
- Fixed Linear Interaction

Proposições

- Estratégia Híbrida Centralizada-Distribuída

Proposições

- Estratégia Híbrida Centralizada-Distribuída
- Minimizar o custo de convergência minimizando o número de vértices do grafo

Proposições

- Estratégia Híbrida Centralizada-Distribuída
- Minimizar o custo de convergência minimizando o número de vértices do grafo
- Ganho de $\Omega(\log n)$ no tempo

Proposições

- Estratégia Híbrida Centralizada-Distribuída
- Minimizar o custo de convergência minimizando o número de vértices do grafo
- Ganho de $\Omega(\log n)$ no tempo

Implementação

- Consiste em criar grupos de nós que ajam como um só.
- Necessita de um passso de formação de clustes inexistente nos algoritmos tradicionais
- Análise do ganho é feita considerando o número máximo de clusters possiveis dependente de r no G (n,r(n))

Requisitos

É assumido uma formação de *clusters* seguindo 2 princípios:

- Cada nó pertence a a um e somente um cluster

Requisitos

É assumido uma formação de *clusters* seguindo 2 princípios:

- Cada nó pertence a a um e somente um cluster
- Em cada cluster existe um nó cabeça que é adjacente a todos os outros nós remanescentes

Requisitos

É assumido uma formação de *clusters* seguindo 2 princípios:

- Cada nó pertence a a um e somente um cluster
- Em cada cluster existe um nó cabeça que é adjacente a todos os outros nós remanescentes
- A distância entre os nós cabeça é $r < d < 3r$

Requisitos

É assumido uma formação de *clusters* seguindo 2 princípios:

- Cada nó pertence a a um e somente um cluster
- Em cada cluster existe um nó cabeça que é adjacente a todos os outros nós remanescentes
- A distância entre os nós cabeça é $r < d < 3r$

Etapas

O Processo de inicialização do modelo pré-consenso tem as seguintes etapas

1. Formação do cluster, nós se identificam como *HEAD* (Cabeça) ou do tipo *CHILD*

Etapas

O Processo de inicialização do modelo pré-consenso tem as seguintes etapas

1. Formação do cluster, nós se identificam como *HEAD* (Cabeça) ou do tipo *CHILD*
2. Detecção de vizinhos, onde os nós cabeça encontram os clusters adjacentes através dos nós gateway

Etapas

O Processo de inicialização do modelo pré-consenso tem as seguintes etapas

1. Formação do cluster, nós se identificam como *HEAD* (Cabeça) ou do tipo *CHILD*
2. Detecção de vizinhos, onde os nós cabeça encontram os clusters adjacentes através dos nós gateway
3. Consenso local, onde nós cabeça computam o consenso dos filhos e repassam o resultado

Etapas

O Processo de inicialização do modelo pré-consenso tem as seguintes etapas

1. Formação do cluster, nós se identificam como *HEAD* (Cabeça) ou do tipo *CHILD*
2. Detecção de vizinhos, onde os nós cabeça encontram os clusters adjacentes através dos nós gateway
3. Consenso local, onde nós cabeça computam o consenso dos filhos e repassam o resultado

Formação de Cluster

1. *CLUSTER_INIT*: onde cada um dos nós decrementa o seu contador interno até o limite 0 para se decidir se será cabeça ou filho.

Formação de Cluster

1. *CLUSTER_INIT*: onde cada um dos nós decrementa o seu contador interno até o limite 0 para se decidir se será cabeça ou filho.
2. Os nós cabeça enviam uma mensagem *CLUSTER_INIT* para os nós adjacentes e espera pelas respostas

Formação de Cluster

1. *CLUSTER_INIT*: onde cada um dos nós decrementa o seu contador interno até o limite 0 para se decidir se será cabeça ou filho.
2. Os nós cabeça enviam uma mensagem *CLUSTER_INIT* para os nós adjacentes e espera pelas respostas
3. Os nós que recebem o pedido de inicialização respondem com uma intenção de se unir ao cluster com uma mensagem *CLUSTER_JOIN*

Formação de Cluster

1. *CLUSTER_INIT*: onde cada um dos nós decrementa o seu contador interno até o limite 0 para se decidir se será cabeça ou filho.
2. Os nós cabeça enviam uma mensagem *CLUSTER_INIT* para os nós adjacentes e espera pelas respostas
3. Os nós que recebem o pedido de inicialização respondem com uma intenção de se unir ao cluster com uma mensagem *CLUSTER_JOIN*
4. O nó cabeça devolve uma mensagem *CLUSTER_ACCEPTED* para o nó filho e adiciona-o a sua lista de filhos.

Formação de Cluster

1. *CLUSTER_INIT*: onde cada um dos nós decrementa o seu contador interno até o limite 0 para se decidir se será cabeça ou filho.
2. Os nós cabeça enviam uma mensagem *CLUSTER_INIT* para os nós adjacentes e espera pelas respostas
3. Os nós que recebem o pedido de inicialização respondem com uma intenção de se unir ao cluster com uma mensagem *CLUSTER_JOIN*
4. O nó cabeça devolve uma mensagem *CLUSTER_ACCEPTED* para o nó filho e adiciona-o a sua lista de filhos.

Após a formação um nó pode se do tipo *HEAD*(Cabeça) ou do tipo *CHILD* (Nó a um *hop* de um *cluster* podendo ao mesmo tempo ser *gateway* . O processo de formação pode ser dividido em 3 etapas:

Formação de Cluster

1. *CLUSTER_INIT*: onde cada um dos nós decrementa o seu contador interno até o limite 0 para se decidir se será cabeça ou filho.
2. Os nós cabeça enviam uma mensagem *CLUSTER_INIT* para os nós adjacentes e espera pelas respostas
3. Os nós que recebem o pedido de inicialização respondem com uma intenção de se unir ao cluster com uma mensagem *CLUSTER_JOIN*
4. O nó cabeça devolve uma mensagem *CLUSTER_ACCEPTED* para o nó filho e adiciona-o a sua lista de filhos.

Após a formação um nó pode se do tipo *HEAD*(Cabeça) ou do tipo *CHILD* (Nó a um *hop* de um *cluster* podendo ao mesmo tempo ser *gateway* . O processo de formação pode ser dividido em 3 etapas:

Detecção de vizinhos

1. Nó cabeça envia um pedido de procura de vizinhos para os seus filhos que repassam por broadcast para seus adjacentes

Detecção de vizinhos

1. Nó cabeça envia um pedido de procura de vizinhos para os seus filhos que repassam por broadcast para seus adjacentes
2. Ao receber um pedido o nó avalia se é do seu cluster e descarta a mensagem

Detecção de vizinhos

1. Nó cabeça envia um pedido de procura de vizinhos para os seus filhos que repassam por broadcast para seus adjacentes
2. Ao receber um pedido o nó avalia se é do seu cluster e descarta a mensagem
3. Se não é do mesmo o nó salva o caminho (link) por onde veio a mensagem e repassa para o nó cabeça

Detecção de vizinhos

1. Nó cabeça envia um pedido de procura de vizinhos para os seus filhos que repassam por broadcast para seus adjacentes
2. Ao receber um pedido o nó avalia se é do seu cluster e descarta a mensagem
3. Se não é do mesmo o nó salva o caminho (link) por onde veio a mensagem e repassa para o nó cabeça

Computação local

1. Nó cabeça envia pedido de $V_{[}0]$ para os seus filhos, computa e devolve os dados
2. Cabeça computa e devolve o dado para os filhos

Computação local

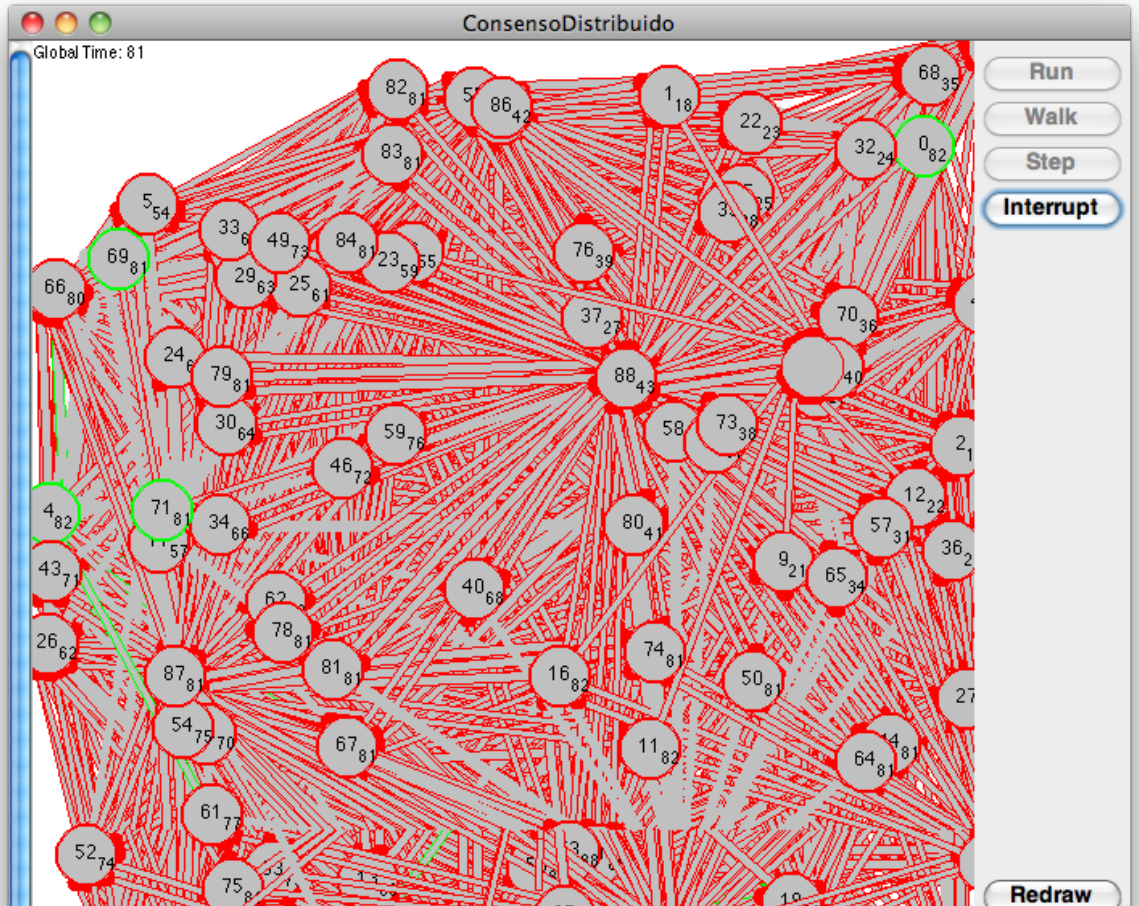
1. Nó cabeça envia pedido de $V_{[}0]$ para os seus filhos, computa e devolve os dados
2. Cabeça computa e devolve o dado para os filhos

Tempos de Espera

- A fase de procura de vizinhos só pode iniciar ao término global da formação dos cluters
- A computação local independente mas fim da etapa global depende dos outros
- O tempo de espera é função do número de vizinhos
- O número de arestas é dependente do raio de transmissão

Formação de cluster no DAJ

Implementamos uma simulação do processo de formação de clusters proposto por LI no simulador DAJ *(*Distributet Algorithms in Java).



Parâmetros de execussão

- Executadas instâncias com nós de 10 a 90 com incremento de 10

Parâmetros de execussão

- Executadas instâncias com nós de 10 a 90 com incremento de 10
- Executadas instâncias com raio de 150, 300, 600 e 850

Parâmetros de execussão

- Executadas instâncias com nós de 10 a 90 com incremento de 10
- Executadas instâncias com raio de 150, 300, 600 e 850

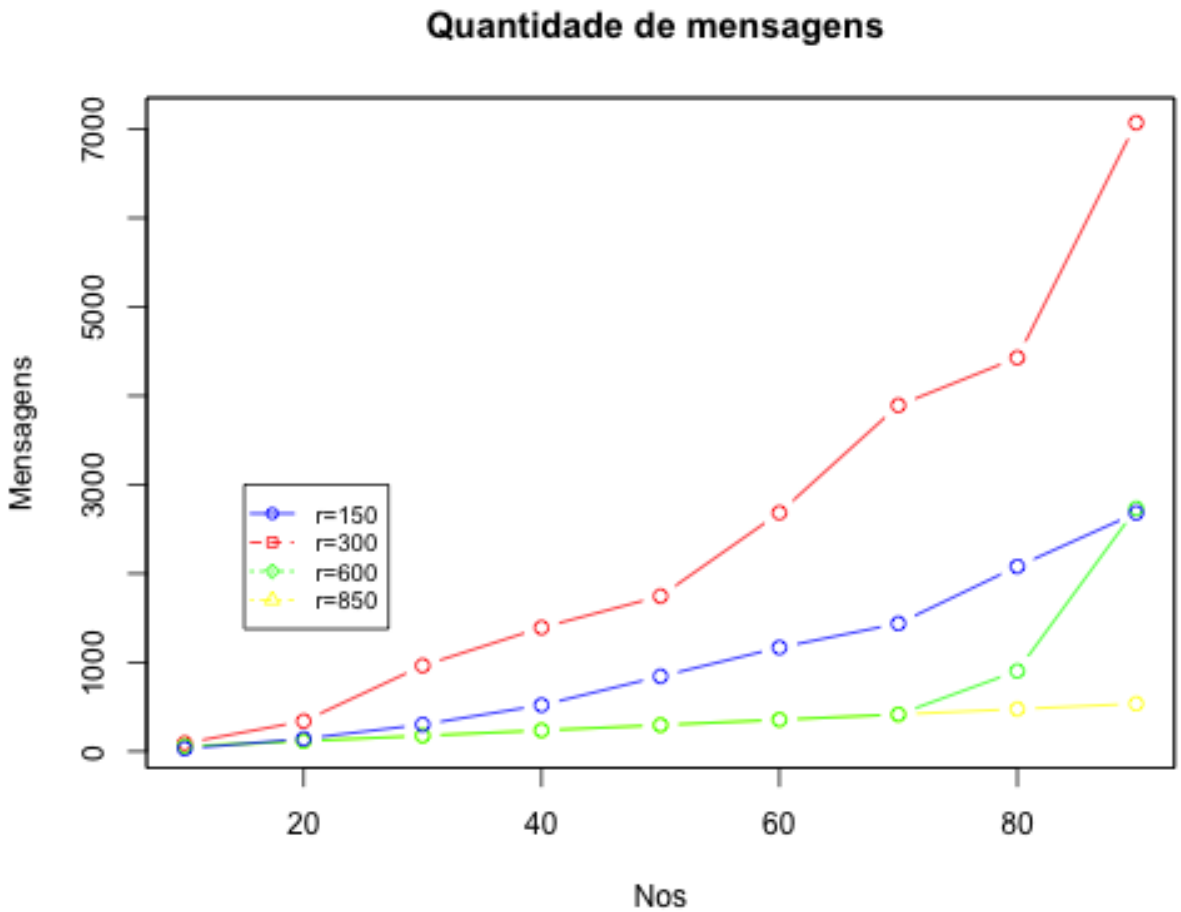


Figura: Mensagens versus Nós considerando o broadcast

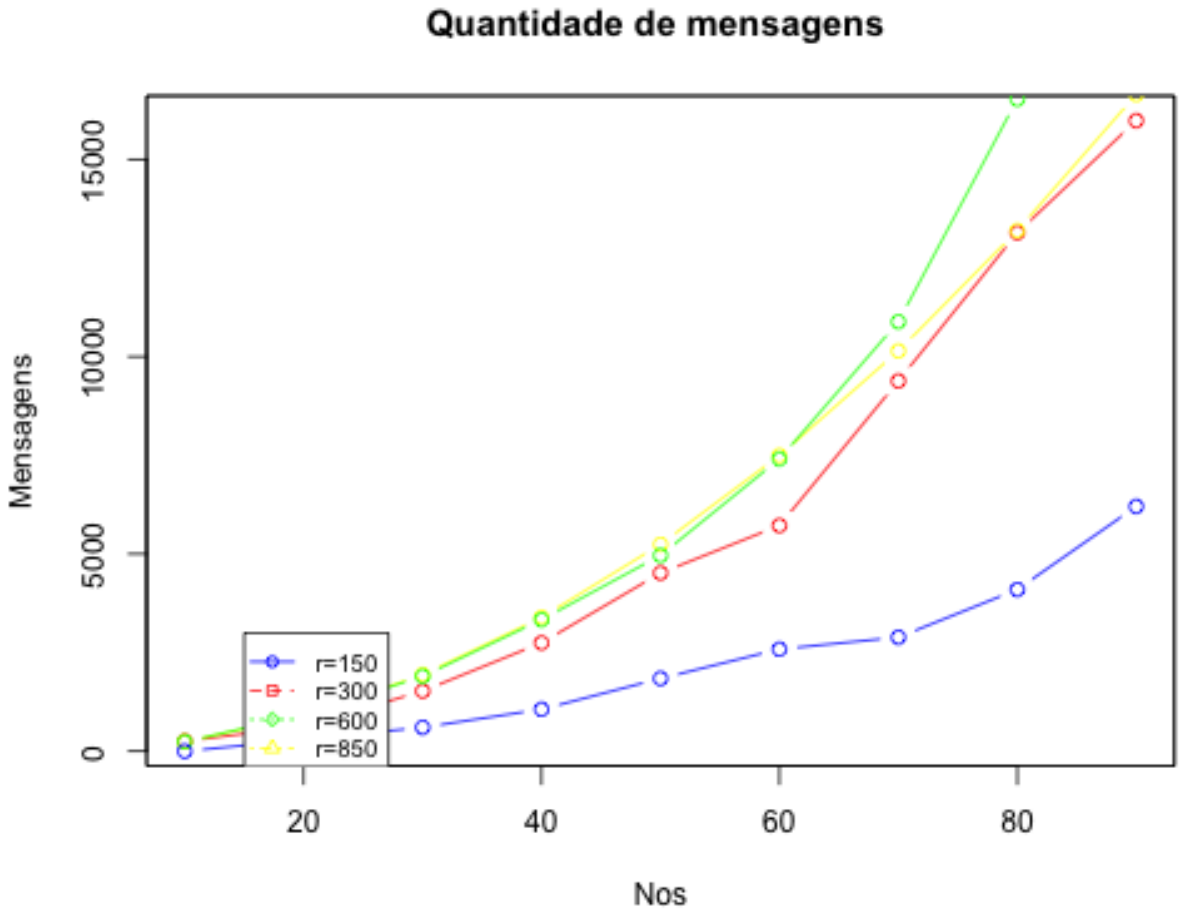


Figura: Mensagens versus Nós desconsiderando broadcast broadcast

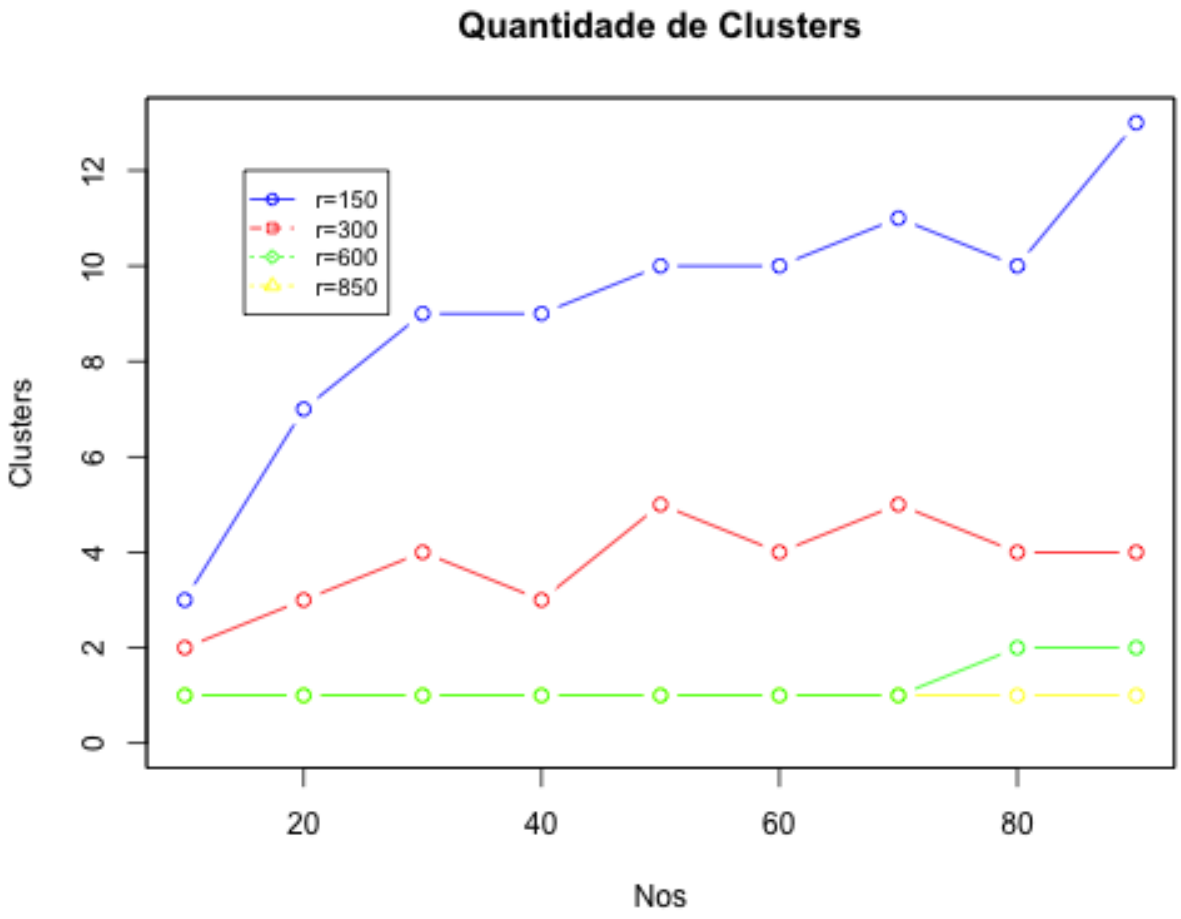


Figura: Nós versus Clusters formados

Conclusões sobre os resultados

- A formação de cluster gera uma quantidade consideravel de mensagens na rede

Conclusões sobre os resultados

- A formação de cluster gera uma quantidade consideravel de mensagens na rede
- Se considerarmos a recepção de mensagens a complexidade é de $O(n^2)$

Conclusões sobre os resultados

- A formação de cluster gera uma quantidade consideravel de mensagens na rede
- Se considerarmos a recepção de mensagens a complexidade é de $O(n^2)$
- O tempo de formação de cluster é quadraticamente proporcional ao número de nós no cluster

Conclusões sobre os resultados

- A formação de cluster gera uma quantidade consideravel de mensagens na rede
- Se considerarmos a recepção de mensagens a complexidade é de $O(n^2)$
- O tempo de formação de cluster é quadraticamente proporcional ao número de nós no cluster
- Como a computação só pode iniciar após a formação os clusters tem de esperar um tempo $O(n^2)$

Conclusões sobre os resultados

- A formação de cluster gera uma quantidade consideravel de mensagens na rede
- Se considerarmos a recepção de mensagens a complexidade é de $O(n^2)$
- O tempo de formação de cluster é quadraticamente proporcional ao número de nós no cluster
- Como a computação só pode iniciar após a formação os clusters tem de esperar um tempo $O(n^2)$

Trabalhos futuros

- Propor formas de balancear o tempo de espera de formação do cluster
- Estudo de outras formas de construção de cluster
- Implementação de ambiente real

W. Li and H. Dai, “Cluster-based distributed consensus,” *Trans. Wireless. Comm.*, vol. 8, pp. 28–31, January 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1657616.1657622

L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” *Systems Control Letters*, vol. 53, no. 1, pp. 65 – 78, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167691104000398

S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, “Randomized gossip algorithms,” *Information Theory, IEEE Transactions on*, vol. 52, no. 6, pp. 2508 – 2530, june 2006.