QoS Control for Sensor Networks

Ranjit Iyer
Leonard Kleinrock

Aluno: Eduardo Hargreaves

Professor: Luís Felipe M. de Moraes

Introdução

- As Redes de Sensores têm características bem particulares entre elas:

- Alta densidade de elementos
- Restrições severas em relação ao consumo de energia
- Dados redundantes
- Muitos fluxos de informação
- Os nós trabalham de uma forma cooperativa

Introdução

- Este processamento distribuído permite uma visão mais completa do ambiente analisado
- Então QoS em uma rede de sensores seria: Encontrar o número ideal de sensores que forneçam informações suficientes para um dado fenômeno observado

Introdução

- Importante por dois motivos:
  - Maximiza a vida útil dos sensores
  - Permite uma suficiente troca de informação entre os sensores

Dificuldades

- As posições e o número de sensores não são necessariamente pré-determinados
- Os sensores morrem (por falta de bateria) e nascem (são repostos) sem muito controle

Objetivo

- Então o objetivo do artigo é propor um algoritmo que baseado na resolução dos dados observados permita que o número ótimo de sensores seja encontrado pela estação base com a ajuda do Gur Game

Gur Game

- Inventado por M.L. Tselin
- Suponha que existam n jogadores, jogando de forma independente
- A cada segundo um juiz pergunta a cada jogador se ela vai votar “sim” ou “não” e conta o número de votos “sim”
- Uma função de recompensa r(k) é gerada em função do número k de jogadores que votaram sim

Gur Game

- Cada jogador é recompensado com probabilidade r(k) e penalizado com probabilidade 1- r(k) independente do seu voto mas dependente do número de pessoas que votaram “sim”
- Todo jogador joga sempre tentando maximizar sua probabilidade de recompensa

Gur Game

- Seja k* o número ideal de jogadores que votem sim
- Pode-se mostrar que independente do número de jogadores pode-se “construí-los”de tal forma que aproximadamente k* jogadores votem sim após um número suficiente de tentativas

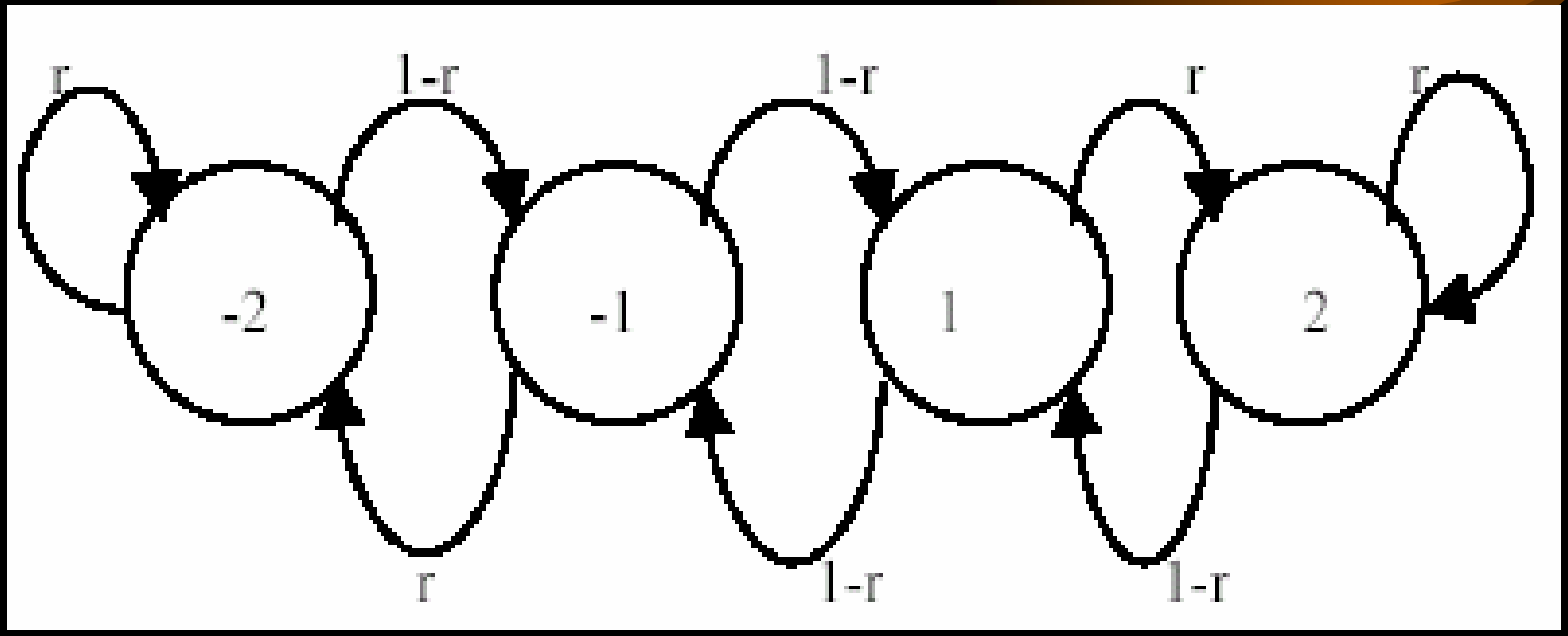
Implementação

- Uma pergunta natural é: Como vou criar estes jogadores?
- Criando em cada jogador uma memória dos seus votos anteriores
- Esta memória é criada usando-se uma cadeia de Markov onde cada jogador vota de acordo com seu estado

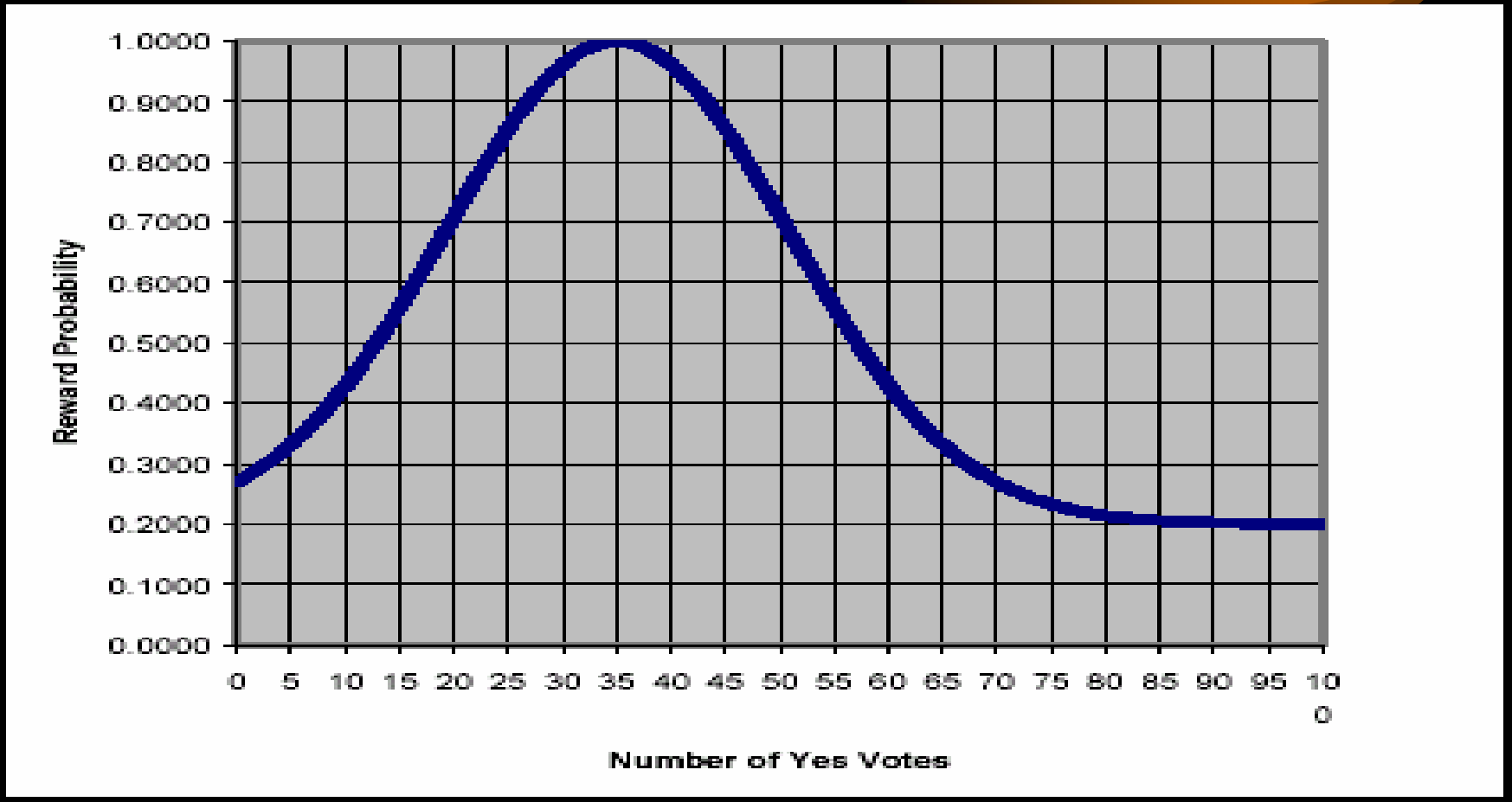
Implementação

- A cadeia de Markov varia de um estado -N até um estado N
- Cada jogador vota de acordo com seu estado
- Se você é recompensado você se move para as extremidades
- Se você é penalizado você se move para o centro

Cadeia de Markov com N=2



Típica função de recompensa(k=35)*



Gur game - rede de sensores

- Cada sensor vota sim se estiver em um estado positivo e não se estiver em um negativo
- “sim”significa sensor ligado
- É muito difícil saber o número atual de sensores ligados então faz-se uma estimativa baseada no número de pacotes recebidos

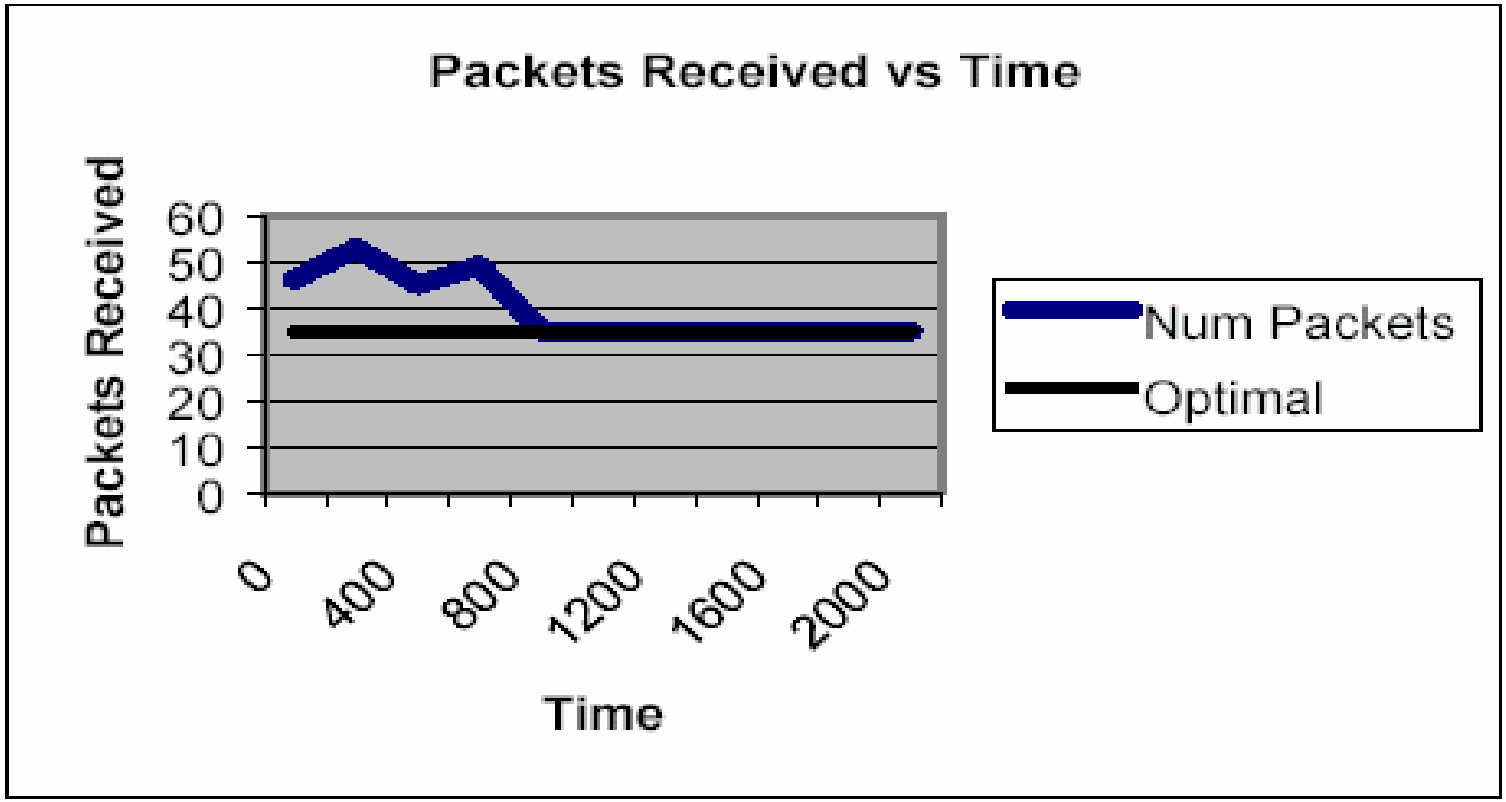
Gur game - rede de sensores

- A estação base conta o número k` de pacotes recebidos
- Então determina r(k`)
- Transmite esta informação a todos os sensores
- Cada sensor se recompensa com probabilidade r(k`)

Resultados-Simulação 1

- A memória tem tamanho N=1
- O número de sensores é constante (100)
- O estado inicial de cada sensor é aleatório
- Seja k* = 35
- A função de recompensa é = $0.2 + 0.8^{v}$ onde $v = -0.002(k_t-35)^2$
- k_t número de pacotes recebidos

Resultados-Simulação 1



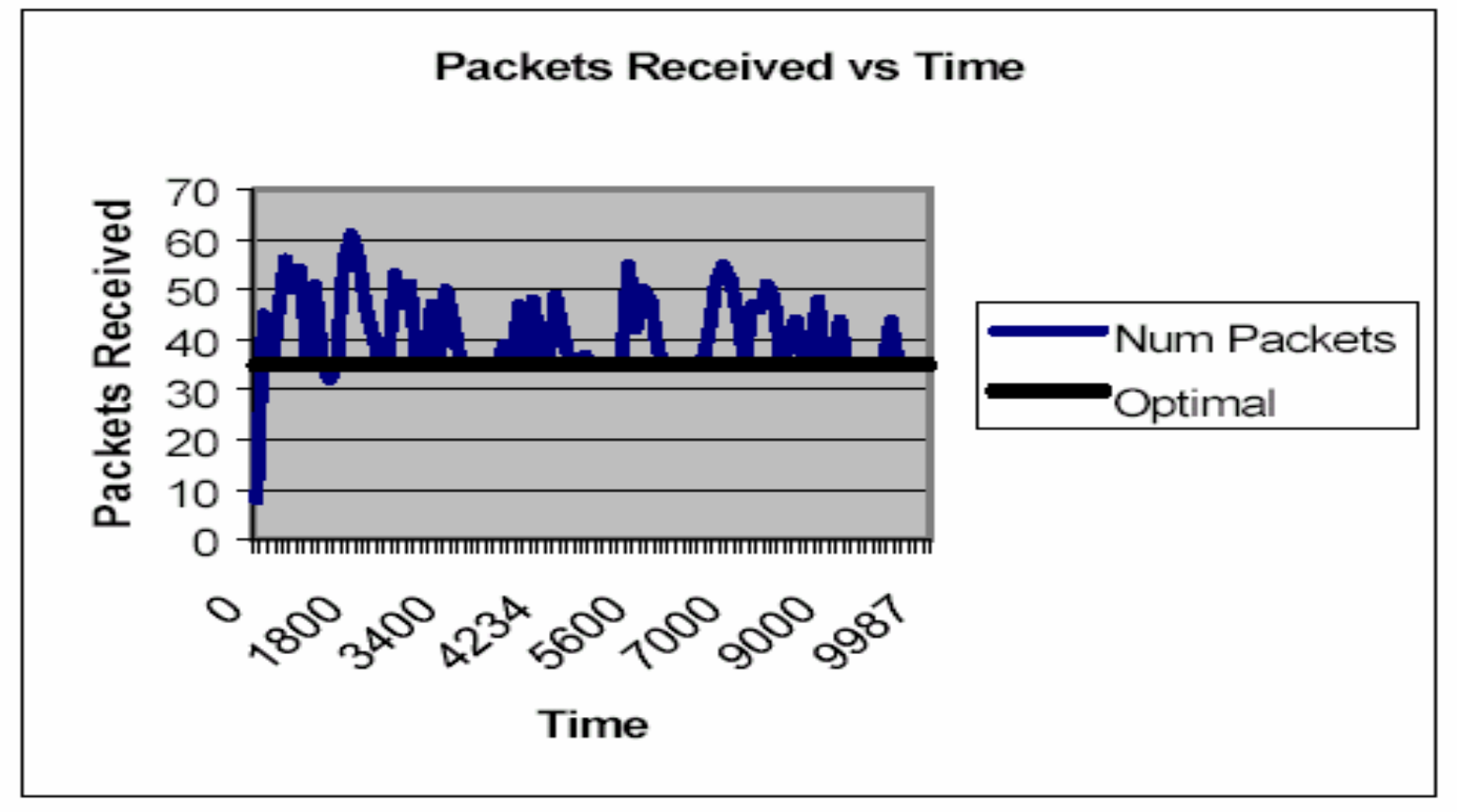
Refinamentos no modelo

- Seja d_i a distância do sensor i a estação base - Espera-se que esta variação aumente o tempo de convergência da algoritmo
- Variações no número de sensores - Espera-se que o número de sensores se estabilize no valor ótimo por períodos da ordem do intervalo entre nascimentos e mortes

simulação 2

- Todos os parâmetros anteriores são mantidos exceto:
- Di: uniformemente distribuído entre 0 e 5
- Intervalo entre nascimentos: exponencial com média 100s
- Tempo de vida: exponencial com média 101s
- População inicial=100

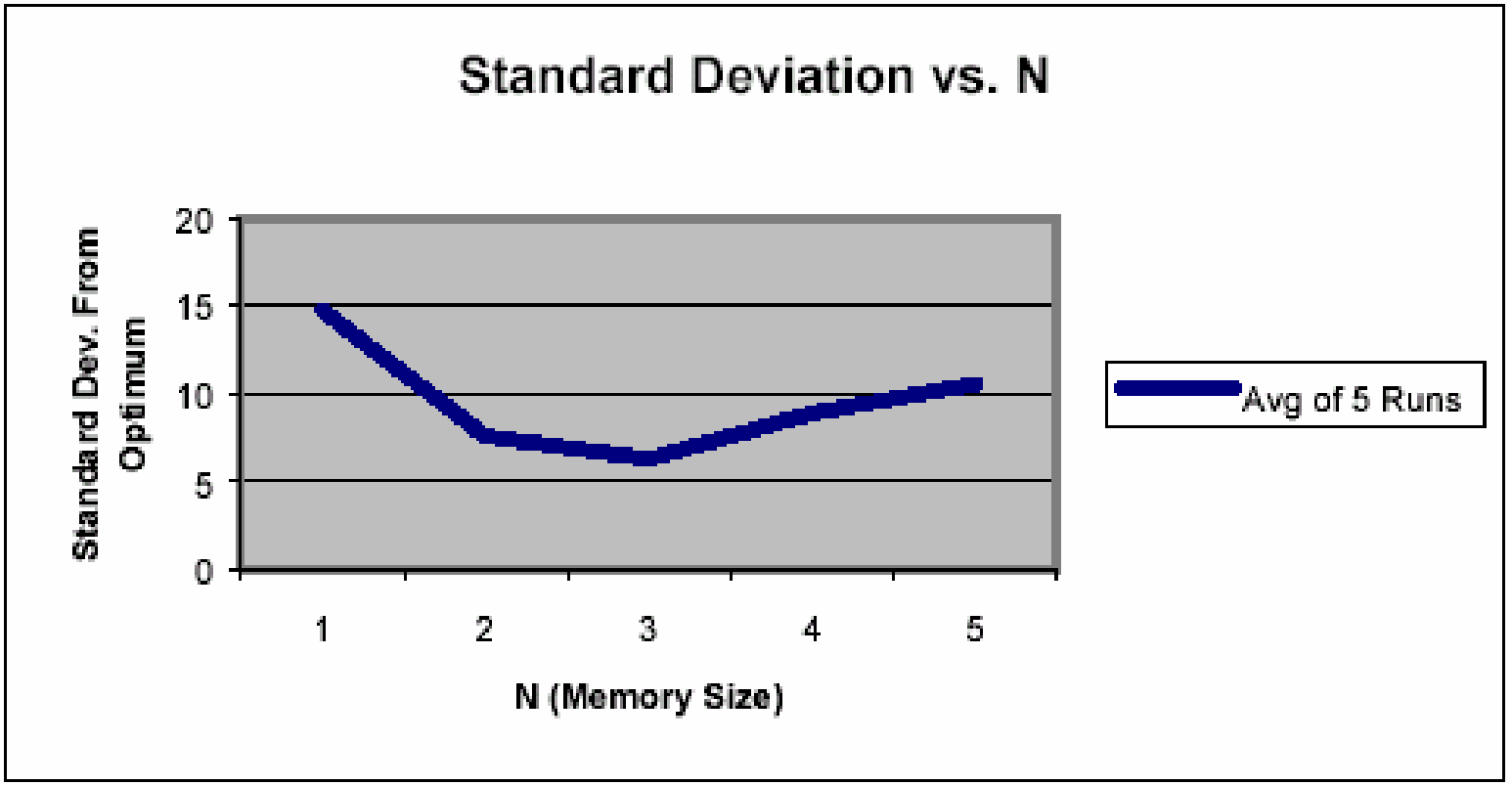
Resultados - simulação 2



Simulação 3

- Verificar o efeito da memória N
- Todos os parâmetros da simulação 2 são mantidos exceto o valor de N que é variável

Resultados-Simulação 3



Conclusões

- Este algoritmo é importante porque permite que a estação base especifique o número ótimo de sensores ativos(tarefa não trivial)
- Com isso a estação base é capaz de ajustar a qualidade de serviço em função do meio ambiente analisado

Referências

- [1] R. Iyer, L.Kleinrock “QoS Control for Sensor Networks” IEEE International conference on Communications , vol.1, 2003 , pag: 517 a 521
- [2] I . F.akyildiz, W . Su, Y . Sankarasubramaniam, E. Cayirci “ A Survey on Sensor Networks” IEEE Comm. Magazine, agosto 2002, pag: 102 a 114