Stochastic List Update Seminar

Loric ANDRE

05.07.2021

Outline

Introduction

- Add dependencies to classic list update problem
- Current goal : Stochastic case, probabilities associated with nodes
- Based on Rivest's "On Self-Organizing Sequential Search Heuristics"

Rivest Paper

- No dependencies yet
- Competitive ratio for Move To Front

Model

- n nodes, $1, ..n$
- Request probabilities $p_1, ..p_n$
- Cost of any algorithm ALG : $C_{ALG} = \sum_{i=1}^{n} pos(i) \cdot p_i$

MTF

- Move To Front
- Move accessed node to the front of the list

Cost of MTF

- $pos(i) = |\{\text{nodes in front of node i}\}| + 1$
- $b(i, j)$ probability that node i is in front of node j
- Average number of nodes in front of node j : $\sum_{i \neq j} b(i, j)$
- Conclusion : Average asymptotic cost :

$$C_{MTF} = \sum_{j=1}^{n} p_j \cdot \left(1 + \sum_{i \neq j} b(i, j) \right)$$

b(i, j) for MTF

- Node i in front of node j if and only if node i accessed after node j
- At any point, node i accessed then neither node i nor node j anymore
- Conclusion :

$$\begin{aligned} b(i,j) &= \sum_{k=1}^{\infty} p_i \cdot (1 - p_i - p_j)^{k-1} \\ &= \frac{p_i}{p_i + p_j} \end{aligned} \tag{1}$$

Static Optimal Algorithm

- Sort nodes by probability
- If $p_i \geq p_{i+1}$

$$C_{STAT} = \sum_{i=1}^{n} i \cdot p_i \tag{2}$$

Conclusion for Rivest Paper

In the case without dependencies :

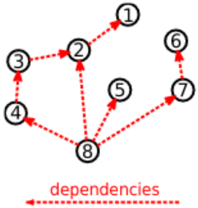
- $b(i,j) = \dfrac{p_i}{p_i + p_j}$
- $C_{MTF} = 1 + \sum_{j=1}^{n} p_j \sum_{i \neq j} \dfrac{p_i}{p_i + p_j} = 1 + \sum_{j=1}^{n} \cdot 2p_j \sum_{1 \leq i < j} \dfrac{p_i}{p_i + p_j}$
- $C_{MTF} \leq 1 + 2 \sum_{j=1}^{n} p_j (j-1)$
- $C_{STAT} = 1 + \sum_{j=1}^{n} p_j (j-1)$
- $\boxed{\dfrac{C_{MTF}}{C_{STAT}} \leq \dfrac{1+2x}{1+x} \leq 2(1 - \dfrac{1}{n+1})}$

Our Work

- Similar reasoning
- Dependencies complicate some steps

Model

- DAG G : dependency graph
- Edge between i and j means i depends on j

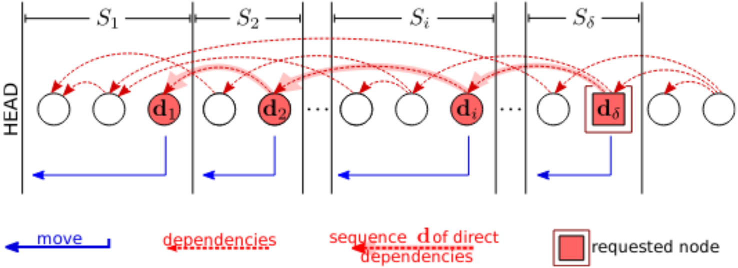


- **Right now : Working with a Forest**

Credits to Juan Vanerio for the figure

Move Recursively Forward

Move accessed node until we meet a dependency, then do the same again with this dependency (recursively) until we reach the head of the list



Credits to Juan Vanerio for the figure

Cost of MRF

- Formula for access cost in terms of b(i, j) holds with dependencies !
- Updated b(i, j)
- b(i, j) = 1 if j depends on i
- b(i, j) = 0 if i depends on j
- Other cases in next slides

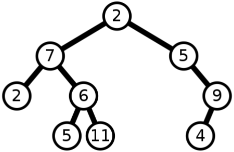
b(i, j) for independent nodes (1/4)

Conditions, at some point :

- i moved : i or a node depending on it accessed
  and
- i moved ahead of j
  and
  - i and j not moved
    or
  - i was not moved and j was not moved enough to overtake i

b(i, j) for independent nodes (2/4) : i moved ahead of j

- i moved if and only if i or a node depending on it is accessed : $\sum_{k \in \mathcal{D}(i)} p_k = \gamma_i$



- i could move ahead of j if and only if its parent was in front of j : this is $b(par(i), j)$

Figure source : *wikimedia*

b(i, j) for independent nodes (2/4) : i and j not moved afterwards

The probability that nodes i and j are not moved afterwards is simply :

$$\sum_{k=1}^{\infty}(1-\gamma_i-\gamma_j)^(k-1) = \frac{1}{\gamma_i+\gamma_j}$$

b(i, j) for independent nodes (3/4) : i not moved and j didn't overtake i

- Like in the previous slide, i not moved is $\frac{1}{\gamma_i}$
- j not overtaking i : j was moved less than m times, m is the number of parents of j after i
- m is between 0 and δ_j (depth of j in its dependency tree)
- Total :

$$\sum_{m=1}^{\delta_j} b(i, par^m(j))(\gamma_j)^m$$

b(i, j) for independent nodes (4/4) : summary

Putting all this together gives us :

$$b(i,j) = \gamma_i \cdot b(par(i),j) \left(\frac{1}{\gamma_i + \gamma_j} + \frac{1}{\gamma_i} \sum_{m=1}^{\delta_j} b(i, par^m(j))(\gamma_j)^m \right)$$

This expression is recursive in both i and j. The b(par(i), j) can be dealt with using a product over the parents but it still leaves the recursion over all parents of j, and as such is not usable in itself.

Cost for STATD

- Static optimal algorithm with dependencies
- **This is the point we're currently having trouble with**
- need a lower bound for its cost
  - Cost higher than STAT (Rivest bound)
  - Node always behind all parents

Conclusion for MRF

- b(i, j) is too complicated in its full form
- Bounding it yields good result in some cases
- We still need a better bound on STATD to get its ratio in all cases

Conclusion

- Currently no numeric bound on the ratio : need a better bound on STATD
- Goal : Less than 4 (current bound for MRF in general case)
- Hope : π in stochastic case

Questions

Thank you for listening, any questions ?