Inspirations from Multiple Race Track Athletics for Multimedia Communications Traffic Engineering: Opportunities and Challenges

Sudip Misra
Yale University

Model

- M racers running towards a goal.
- Non-interfering tracks.
  - If considering interference, probability of interference is important. Overtaking rules should be considered.
- At each instant, racer R_i moves towards the goal with probability s_i.
- Stays where s/he is with a probability $(1 - s_i)$.
- There could be *preferential treatment* (handicaps) given to some racers.
- The results of the race can help *learn* the ordering of the racers.

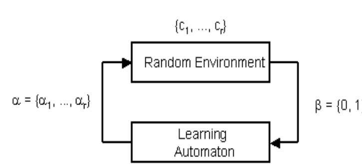
Choosing the best action → Traditional approach

Ordering actions → Random Races approach

3

Traditional Learning

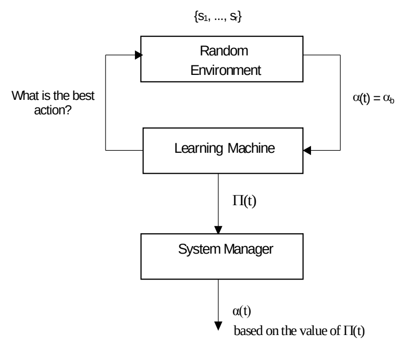
- A learning machine (automaton) is offered a set of actions by a random environment.
- The automaton chooses only one of the offered actions at a time – the action it chooses is based on the action probability vector.
- The environment, which knows the “best action”, either rewards the automaton or penalizes it with a certain penalty probability.
- Based on the continuous interaction between the automaton and the environment, the automaton aims to learn the optimal action, i.e., the action that has the minimum penalty probability.
- This optimal action is eventually chosen more frequently than any other action.



4

Random Races Learning

- Oommen et al.
- LM interacting with an RE.
- RE offers a set of *r* actions.
- LM asks the RE: “which action do you think is the best action?”
- RE suggests α_b with probability s_b.
- Output of the LM is the permutation $\Pi(t)$
- System manager interacts with the real world.
- LM converges to an order of actions $\Pi(\infty)$
- $\Pi(\infty)$ sorted in descending order of their suggestion probabilities.
- As $t \to \infty$, $\Pi(t)$ converges to Π^* with a probability arbitrarily close to unity.



5

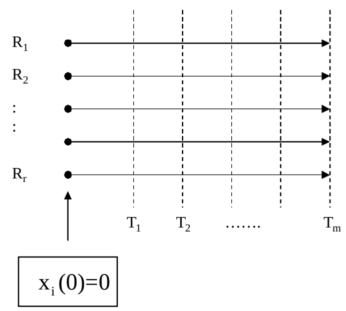
Definition

If $\forall i,j$, where i≠j, the probability ${}_ip_j(t)$ as ${}_ip_j(t) \to 1$ with $t \to \infty$ probability unity whenever $s_i > s_j$, the LM is defined to be *permutationally optimal*. ${}_ip_j(t)$ denotes the probability that at time ‘t’ the automaton picks a permutation in which α_j succeeds α_i

<u>The probability of converging to the best permutation is unity</u>.

6

Multiple Race-Track Learning With No Handicaps



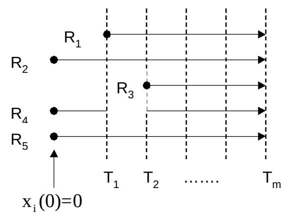
- *r* racers run on multiple non-interfering tracks.
- Racers do not have an *a priori* information of the ordering of the actions at the beginning of the race.
- All the racers start at the same origin.
- The output of this random race is the ordering in which the racers complete the course, i.e., the ordering in which the learning has converged.
- Whenever the Learning Machine asks the Environment about the best action, at that instant the suggestion of the Environment, $\alpha j(t)$, is used by the LM to move Rj one step towards the target (Tm) by incrementing the value of xj(t) by unity.

MRTL With No Handicaps

Theoretical Result

If an LMRT utilizes no *a priori* information in all suggestive random environments, it is permutationally ε*-optimal*

MRTL With Handicaps



- The racers have handicaps at the start of the race
- Racers are provided with *a priori* information before interacting with the environment .
- R1 and R3 do not start at the initial position xi(0).
- An LM may, during the course of the interaction, sometimes find this initial position misleading, and sometimes ignorable

9

MRTL With Uniform Handicaps

- The handicaps are uniformly distributed.
- **Theoretical Result:** If the *a priori* information is ignored by the LM and the racers are given handicaps that are uniformly distributed, the LM will have no way to unlearn the preferential treatment initially provided.

10

Other configurations

- **MRTL With Non-Uniform Handicaps**
  - The starting positions of the racers do not obey a uniform distribution.
- **Single Race-Track Learning**
  - There are *r* racers $\{R_1, R_2, \ldots, R_r\}$ running on a single-race track.

11

Traffic Engineering with MRTL

- Traffic Engineering (TE)
  - Optimize traffic handling and resource utilization on physical network topologies.
- My work (with Oommen and Granmo)
  - Efficient adaptive online routing algorithm
  - Computation of bandwidth-guaranteed paths in TE.
  - Used a MRTL-based learning scheme that computes an optimal ordering of routes.
  - Utilized MRTL for the restricted case when the Environment is "suggestive".
- In a suggestive Environment, at every time instant the Environment provides to the learning mechanism a proposal as to what action it thinks is the best.
- The proposed algorithm was shown to be efficient.

12

Random Races and Traffic Engineering Routing

- Previous algorithms for adaptive TE routing
  - Learning the most suitable action (path) to be taken at any particular time instant.
- MRTL approach
  - Rather than attempting to merely learn the best action/path, it should be advantageous if a learning mechanism could rank the paths in the respective order of their optimality.
- The problem is challenging because the solution space is of size r! as opposed to r as in traditional learning.

13

The MRTL-Based Traffic Engineering Routing Approach

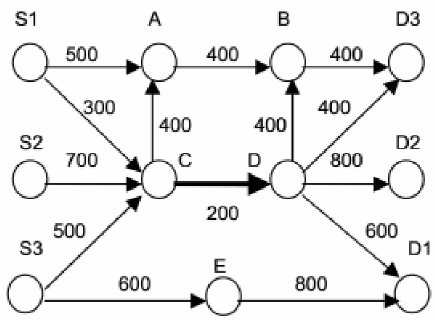
- The learning mechanism has a current understanding of the ranking of the paths.
- It proceeds to choose the paths in the order of their rankings till one accepts the request and permits the traffic to proceed.
- At this juncture, the learning mechanism treats this admission control response as a reward for that particular path.
- It uses this response to update the understanding of the optimal ordering.

14

Critical Links

- Critical links are those that, if congested because of heavy loading, might lead to the rejection of requests.
- The determination of the critical links is based on the concepts of the *maxflow* and the *mincut* computations.
- The *mincut* of an ingress-egress pair of nodes is the set of those edges that, if removed, completely separates the ingress and egress nodes in the pair, and satisfies the property that, of all those sets of edges, it has the minimum cost.
- The critical links are those that, if selected for routing the requests, would lead to the decrease of the *maxflow* values of one or more ingress-egress pairs.

15



A demand of 100 units is to be routed through the ingress-egress pair, (S3, D3).

The critical links for both (S1, D1) and (S2, D2) is {CD}.

All paths that involve the link CD should be discouraged in routing requests.

16

Solution Model

- **The Racers**
  - Station an LM corresponding to every ingress-egress pair of nodes in the graph, whose task is to rank all the outgoing paths (the racers).
  - At every instant, the LM asks the Environment to suggest the best path.
  - The Environment suggests a suitable path from all the possible paths between ingress-egress pair of nodes.
- **The Suggestive Environment**
  - The Environment changes continuously and stochastically.
  - The changes are based on a distribution that is unknown to the Racers, but assumed to be known to the Environment.
  - The Environment suggests an LM with a signal indicative of the best action at a particular time instant.
- **The Feedback - Reward/Penalty**

  The optimality of the paths is inferred using the following information:
  - The number of critical paths
  - The maximum residual bandwidth on a path

17

Environmental Feedback: $\beta(t)$

The feedback corresponding to any suggested action

If C_L is the number of critical links (described below) in the path, and

R is the maximum expected residual bandwidth in the path if the request were routed through that path.

Then $\beta(t) = -k_1 C_L - \frac{k_2}{R}$ where $k_1, k_2 > 0$

18

Algorithm: RRATE

Input

A network with incoming bandwidth routing requests

Output

Requests routed through different paths

Parameters

N: A predefined number, which denotes the maximum number of rewards any of the actions can assume.

BEGIN

Offline Operation

1. Determine the k-shortest paths between each of the IE router pairs.
2. Maintain an RR corresponding to each IE-pair. Each path corresponds to the different actions of the racers.
3. Specify a threshold bandwidth utilization (ρ_{Thresh}) that any link in the network can have at any time instant. This can better be specified as a percentage of the maximum possible bandwidth of a link, rather than a fixed value of bandwidth.

Algorithm: RRATE … Contd.

Online Operation

Pre-convergence

4. Assume that $x_i(n)$ represents the number of rewards received by action *i* at the n^{th} time instant. Initially, set, x_i, $x_i(0)=0$.
5. For each incoming bandwidth setup request for each path between an IE pair, compute beta(t)
6. Select the action with the highest value of beta.
7. If that action is accepted by the Call Admission Control (CAC), and the expected utilization is less than ρ_{Thresh}, then select the action and route the request through that path.
8. Else if the action has an expected utilization greater than the threshold utilization, choose the action with the next highest value of .
9. Increase $x_i \leftarrow x_i + 1$, if the call is accepted for the path i, otherwise repeat Steps 7 and 8 with the next highest value of beta.
10. If all the actions have been tried without any of them being selected, then reject the request.
11. Repeat Steps 5-10 until any one of the values of x_i equals N.
12. Sort the x_i values in the decreasing order of their respective values.

Post-convergence

13. Choose the actions one at a time in the order determined in Step 12.
14. Route requests through the first path of those allowed by the CAC.

Experiments

- **Experiment Set 1:** Comparison of the performance of RRATE with the chosen benchmark algorithms, under different loading conditions, for fixed network size and density.
- **Experiment Set 2:** Comparison of the variation of the performance of RRATE with the chosen benchmark algorithms, and with the variation of the network density.

21

Performance Metrics

- Rejection ratio of requests
- Percentage of accepted bandwidth
- Average route computation time per request

22

Benchmark Algorithms

- Shortest Path Algorithm (SP)
- Shortest Widest Path Algorithm (SWP)
- Widest Shortest Path Algorithm (WSP)
- Minimum Interference Routing Algorithm (MIRA)
- Stochastic Estimator Learning Automata Routing Algorithm (SELA)

23

Rejection ratios

	Number of requests	RRATE	MIRA	SELA	WSP	SWP	SP
Moderate loading	1000	0.368	0.430	0.404	0.447	0.612	0.475
	2500	0.418	0.443	0.421	0.475	0.626	0.491
	5000	0.428	0.439	0.470	0.450	0.619	0.470
Heavy loading	1000	0.568	0.628	0.581	0.625	0.687	0.643
	2500	0.655	0.670	0.668	0.676	0.728	0.683
	5000	0.569	0.580	0.581	0.617	0.678	0.626
Marginal loading	1000	0.069	0.078	0.112	0.150	0.534	0.165
	2500	0.081	0.096	0.128	0.158	0.535	0.184
	5000	0.085	0.108	0.125	0.176	0.558	0.198

24

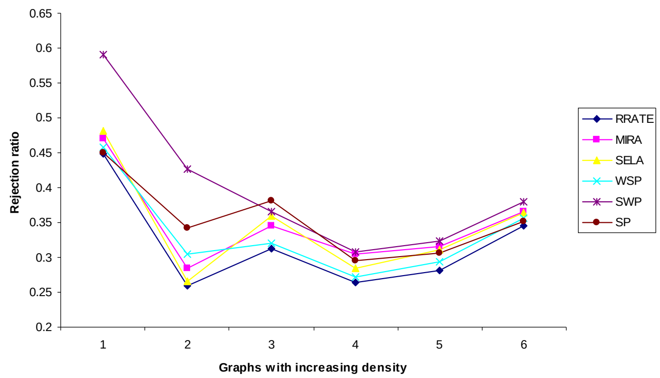
Percentage of Accepted Bandwidth

	Number of requests	RRATE	MIRA	SELA	WSP	SWP	SP
Moderate loading	1000	56.19	51.14	54.46	52.81	35.33	51.14
	2500	54.22	51.10	54.40	50.94	35.85	49.48
	5000	55.07	53.39	51.10	53.13	35.73	53.81
Heavy loading	1000	30.89	26.69	29.84	28.71	25.79	28.01
	2500	29.01	27.89	28.29	29.41	23.74	28.39
	5000	38.01	36.35	38.81	36.08	30.04	35.29
Marginal loading	1000	87.16	86.54	83.54	82.73	44.07	81.30
	2500	88.62	87.59	85.72	83.00	44.05	80.20
	5000	88.72	87.25	86.10	81.10	44.15	79.12

Average route computation time per request

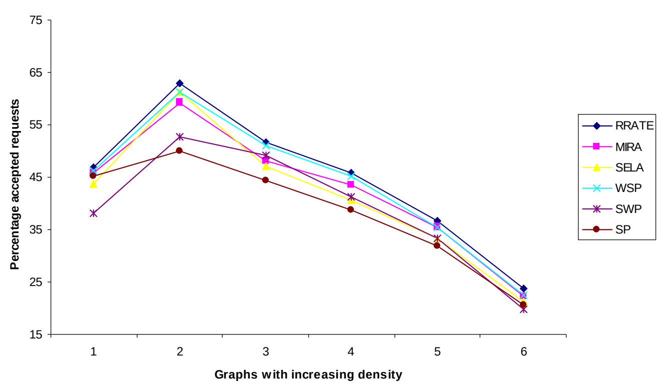
	Number of requests	RRATE	MIRA	SELA	WSP	SWP	SP
Moderate loading	1000	0.003	3.447	0.326	0.004	0.002	0.002
	2500	0.003	3.359	0.326	0.004	0.002	0.002
	5000	0.003	3.555	0.324	0.003	0.002	0.002
Heavy loading	1000	0.003	1.908	0.316	0.002	0.003	0.002
	2500	0.003	1.940	0.307	0.002	0.002	0.002
	5000	0.004	2.794	0.362	0.003	0.002	0.002
Marginal loading	1000	0.003	6.384	0.311	0.002	0.002	0.002
	2500	0.004	6.553	0.349	0.003	0.002	0.002
	5000	0.003	6.572	0.376	0.003	0.003	0.003

Rejection ratio versus graph density



27

Percentage accepted requests vs. graph density



28

Average route computation time per request (in seconds) vs. variation of the network density

Topology ID	RRATE	MIRA	SELA	WSP	SWP	SP
2	0.003	0.236	0.029	0.002	0.002	0.001
3	0.002	0.46	0.039	0.003	0.002	0.002
4	0.003	0.723	0.053	0.004	0.002	0.001
5	0.004	2.126	0.066	0.002	0.003	0.002
6	0.006	2.478	0.068	0.002	0.003	0.003
7	0.005	2.653	0.07	0.003	0.003	0.002

Conclusions

- Powerful computation tool … lot of potential … not popularly known.
- A new class of solutions incorporating the family of stochastic Random-Races (RR) algorithms.
- In contrast to the previously proposed algorithms, the algorithm learns the optimal ordering of the paths through which requests can be routed according to the rank of the paths in the order learnt by the algorithm.
- Better performance.