Modeling TTL-based Internet Caches

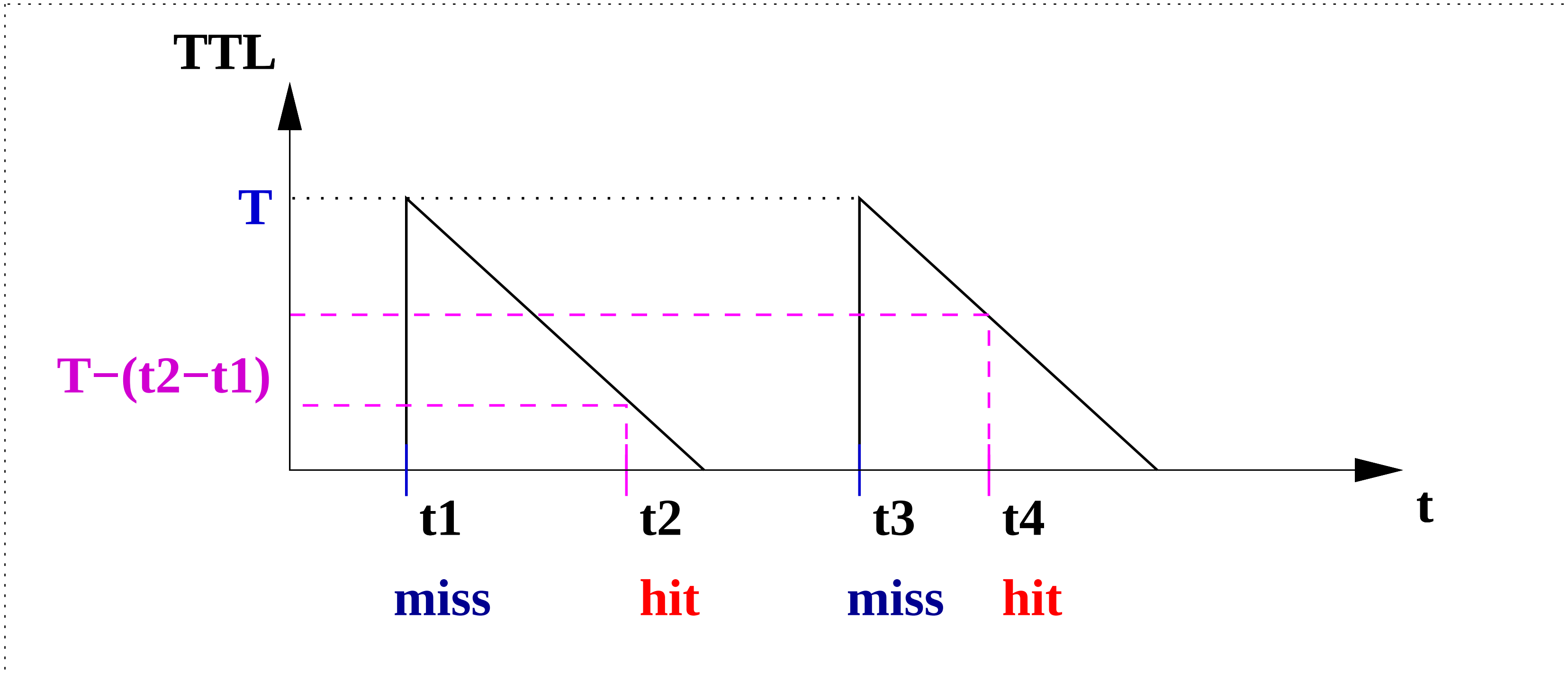
Jaeyeon Jung, Arthur W. Berger , Hari Balakrishnan

MIT LCS

April 2003

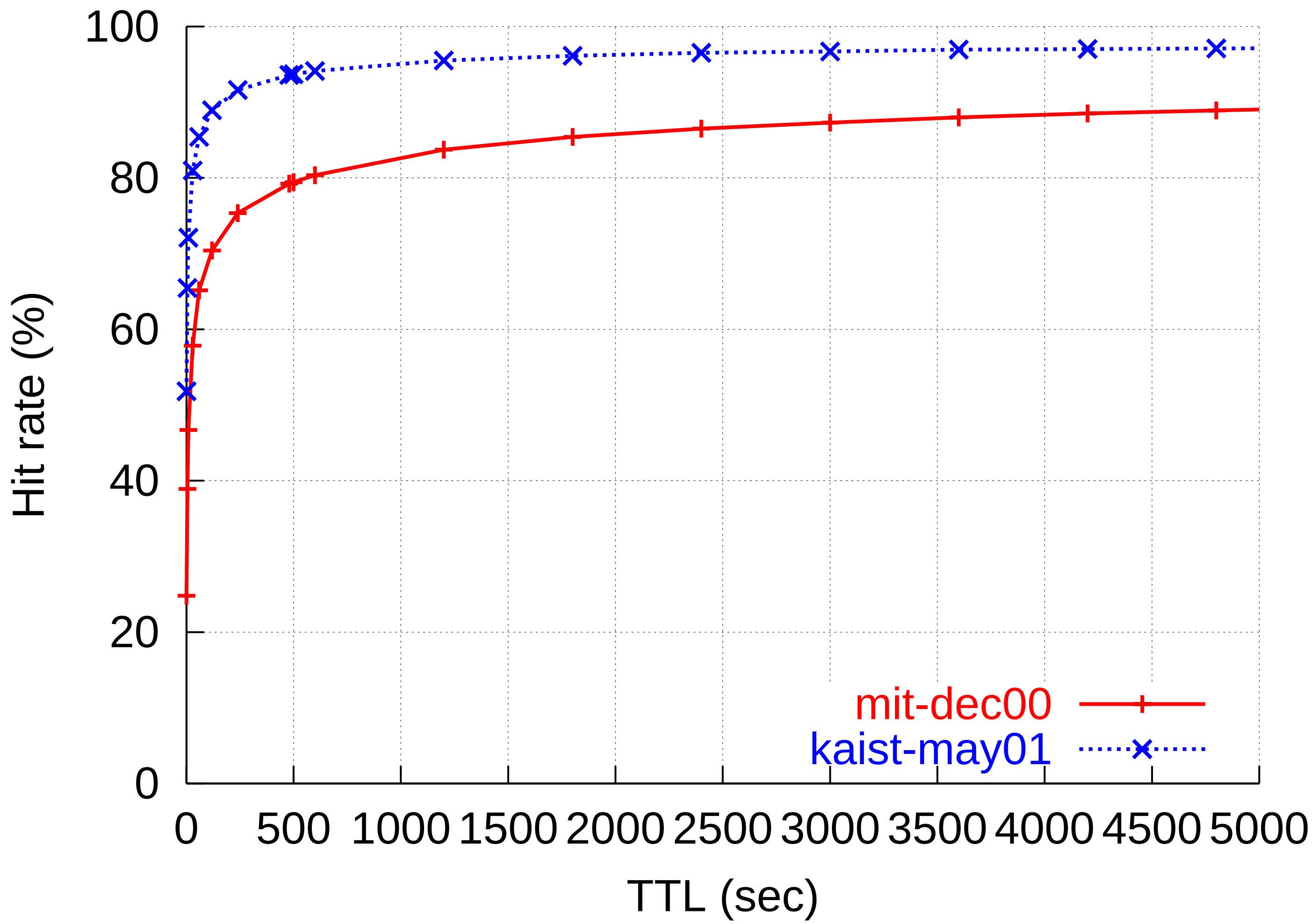
{jyjung,awberger,hari}@lcs.mit.edu

Time-To-Live-based Caches



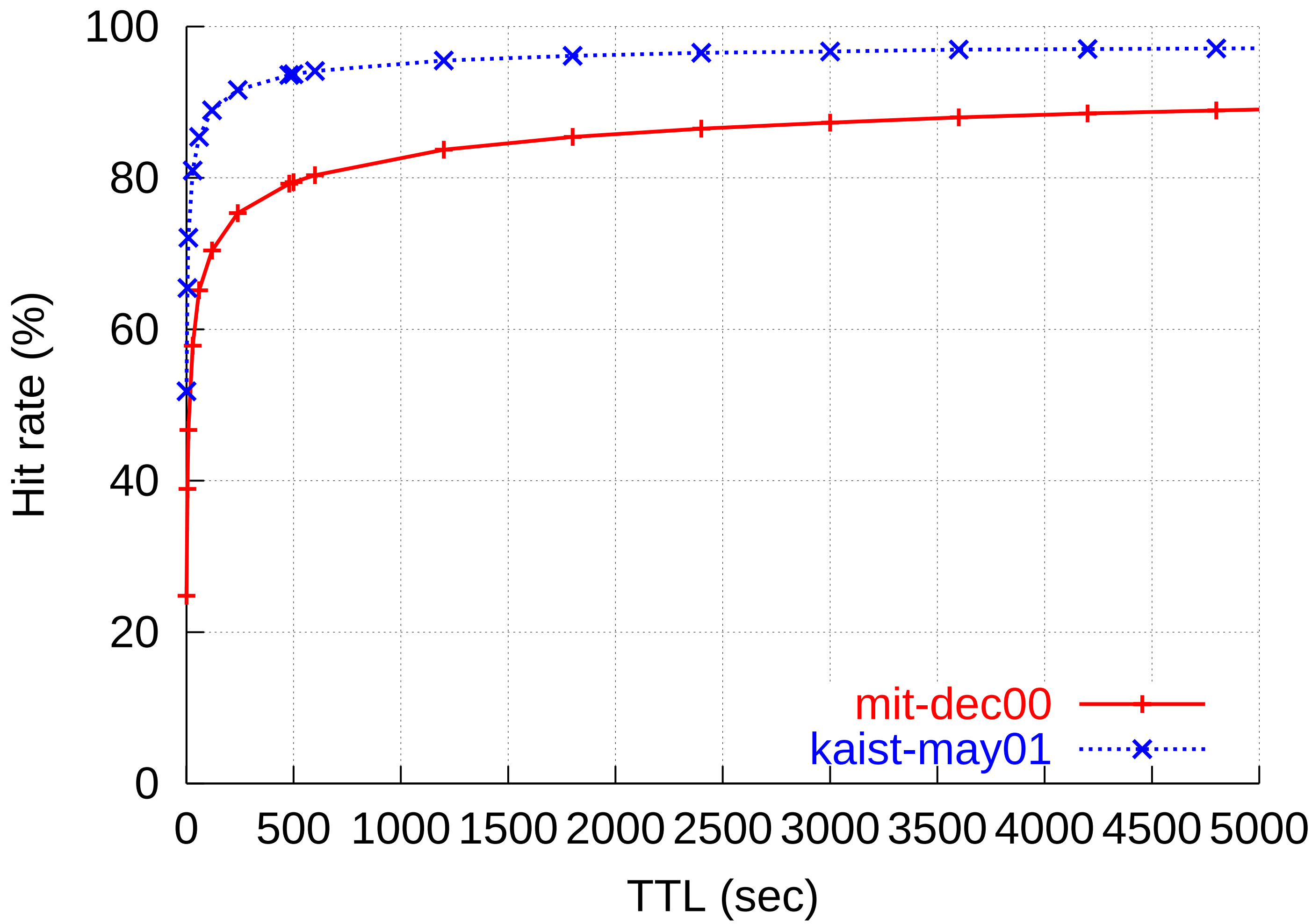
- ✔ Scales well: no need to maintain per requestor states
- ✔ DNS and Web caches
- ✔ Hit rate = $f(\text{TTL, query statistics})$

Motivation



- ✔ DNS cache hit rate rapidly increases as a function of TTL, exceeding 80% for 900 second TTL [JSBM02]

Motivation



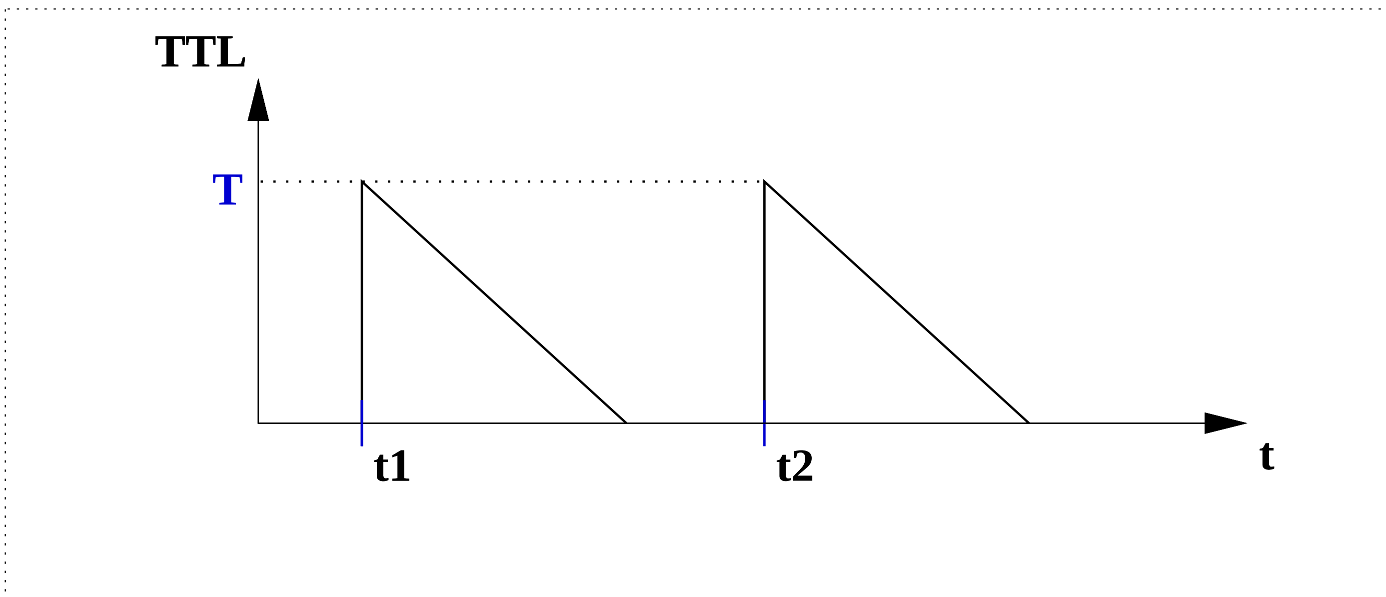
- How does the cache hit rate depend on the statistics of data accesses and the choice of TTL?

Itinerary

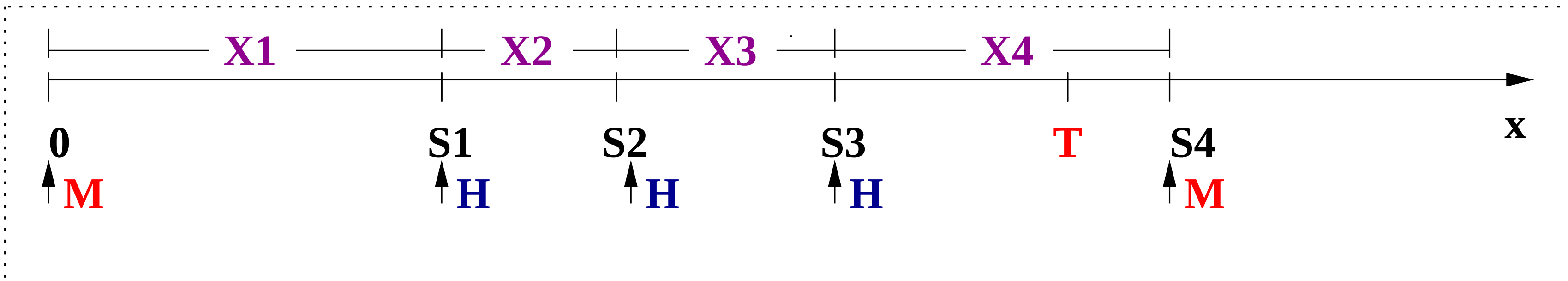
- ✔ Model hit rate as a function of query arrival times and TTL of data items
  - Assumption: cache / query process
  - Formula for hit rates
- ✔ Evaluate the model using real traces
  - Numerical calculation of hit rates
  - Analytic models of inter-query times
  - Comparison of hit rates

Cache Assumption

- ✔ TTL-based consistency control
- ✔ No capacity miss
- ✔ TTL value is always the same for a given data item

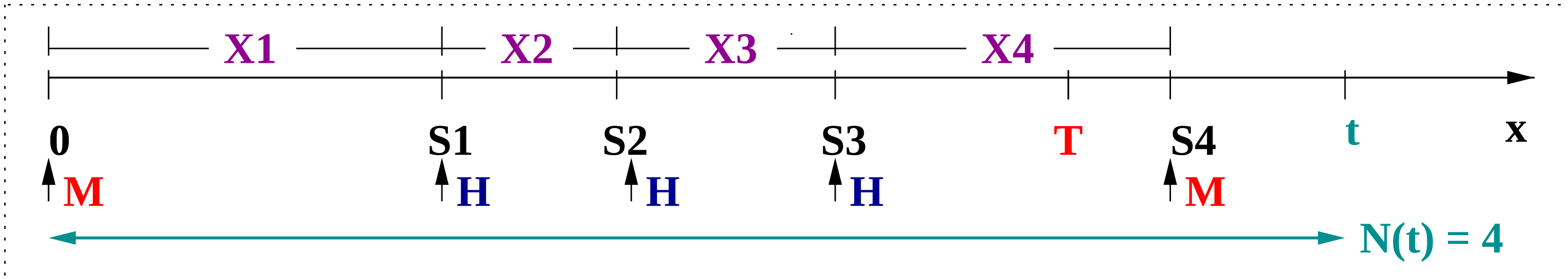


Query Assumption



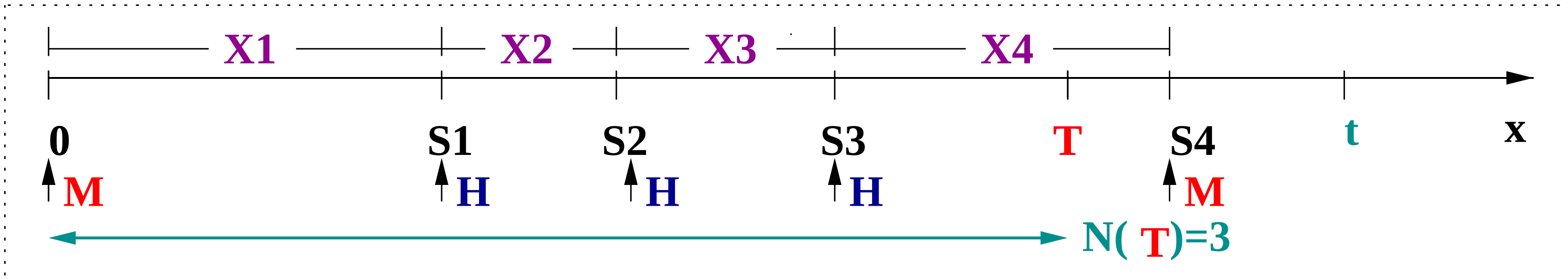
- Let X_i be the time interval between the start time of the i^{th} query and the $i-1^{th}$ query to a given data item
- $X_0 = 0$ and X_i are proper, non-negative, independent and identically distributed (i.i.d.) random variables, X_i may have an infinite mean (renewal assumption)

Notation – $N(t)$



- ✔ Let $N(t)$ equal the number of queries for the given data item in the interval $(0, t]$
- ✔ $N(t)$ is called the renewal counting process

Key Observation



- $N(t)|_{t=T}$ models the number of cache hits per cache miss for a given TTL, T

- $F(t) \equiv \Pr[X_i \le t]$

$$\begin{aligned} \Pr[N(t) \ge n] &= \Pr[S_n \le t] \\ &= \Pr[X_1 + X_2 + \cdots + X_n \le t] \\ &= F^{(n)}(t) \end{aligned}$$

Formula for Hit Rates

- Hit rate $\equiv$ # of hits / # of queries
- $H(u:T) \equiv$ hit rate over the interval $(0,u]$ given the TTL=T
- $H(T) \equiv \lim_{u\to\infty} H(u:T)$

Theorem 1 *If the inter-query times X_i's to a given data item are proper, non-negative, independent and identically distributed random variables, whose mean may be infinite, then*

$$H(T) = \frac{E[N(T)]}{E[N(T)]+1} \text{ with probability one.}$$

Calculation of Hit Rates

- Renewal equation:
  $E[N(t)] = F(t) + \int_0^t E[N(t-x)]dF(x)$
- Discretization yields a numerically convenient iteration of the renewal equation, and thus $H(T)$

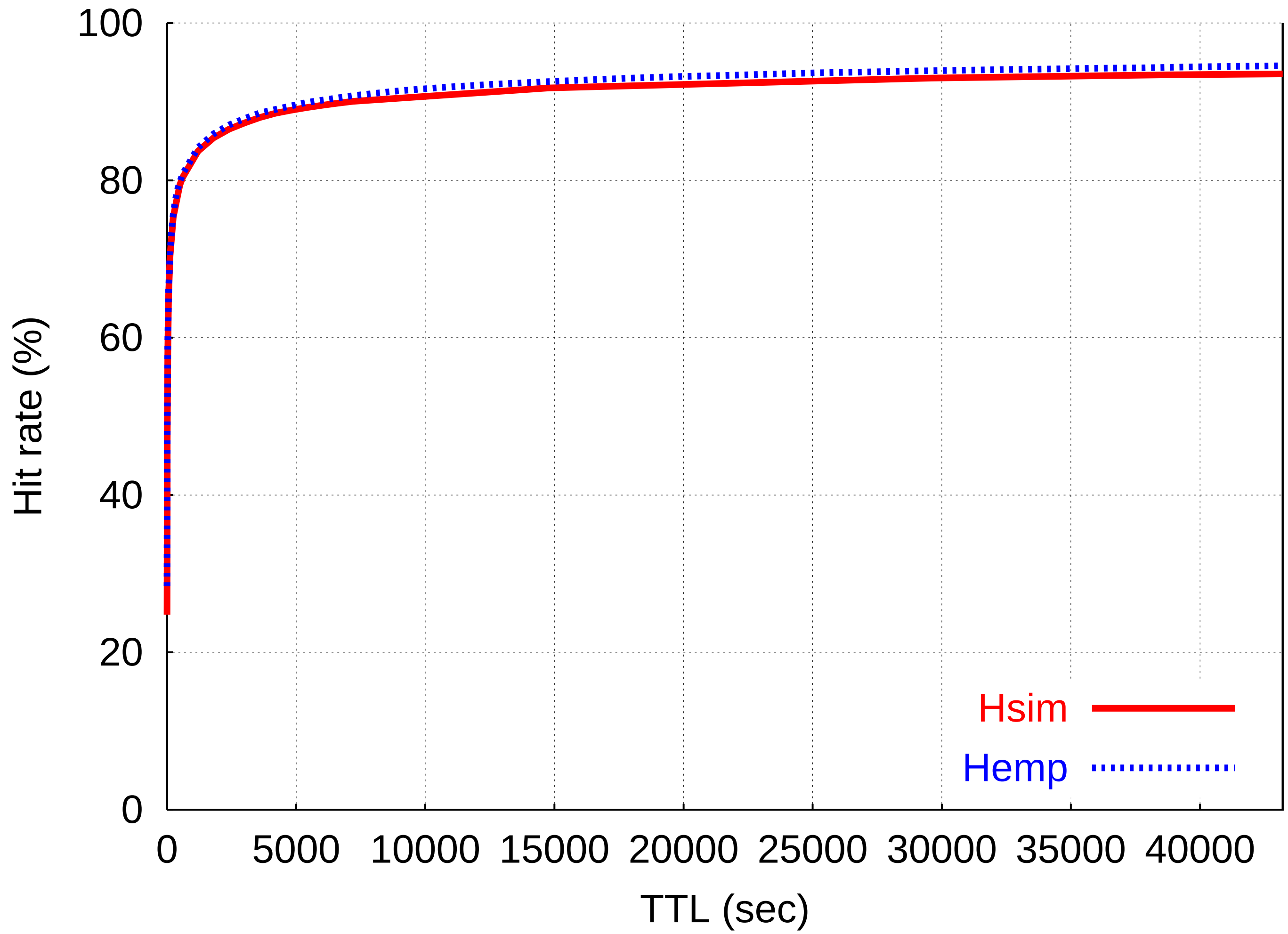
$$H(T) = \frac{E[N(T)]}{E[N(T)]+1}$$

- H_{emp}: renewal assumption with empirical $F(t)$
- H_{ana}: renewal assumption with analytic $F(t)$
- H_{sim}: trace-driven simulation

Numerical Results

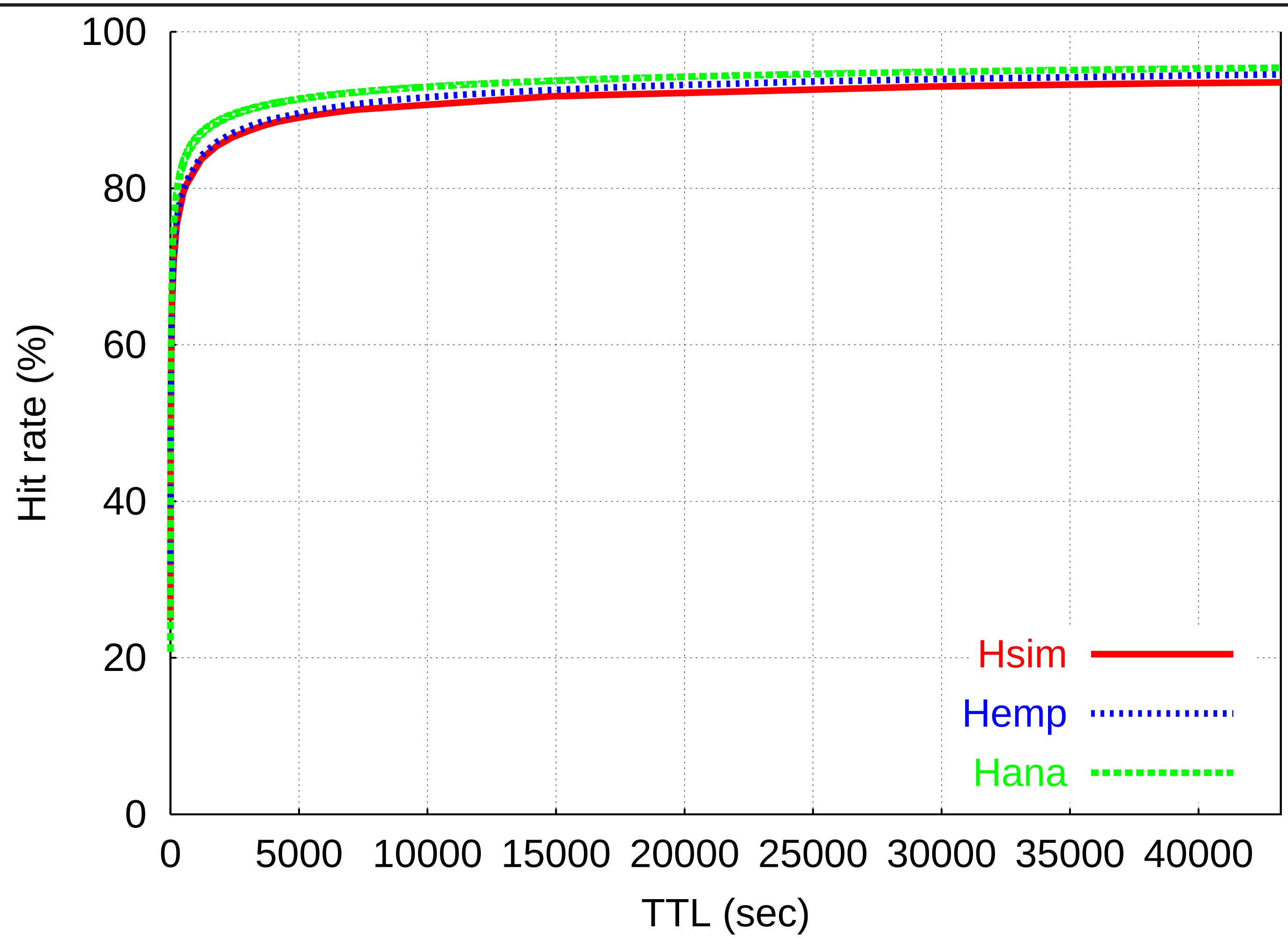
- ✔ Use DNS as an example system, we calculate, H_{emp}, H_{ana}, and H_{sim}
- ✔ Use TCP connection arrivals to model DNS cache references [JSBM02]
  - H_{sim}: trace-driven simulation
  - H_{emp}: Obtain empirical F(t) from the data set
  - H_{ana}: Fit empirical F(t) into a number of well-known probability distributions

Hit Rate Comparison



✔ H_{sim} vs. H_{emp} : the renewal model worked surprisingly well ($\leq 2\%$ difference for TTL in (0,86400] sec)

Hit Rate Comparison



- ✔ H_{ana} is less accurate reflecting the complicated structure of the real inter-query time distribution

Remark

- ✔ For a TTL $T = 900$ sec, hit rates are over 80% for all three traces

- ➔ High variability of inter-query times
  - $H = 80\% \Rightarrow E[N(T)] = 4$ $(H(T) = \frac{E[N(T)]}{E[N(T)]+1})$
  - $E(X) = 2000$ (sec) from real trace
  - If inter-query time distribution $F(t)$ were exponential

$$E[N(T)] = \frac{900}{2000} = 0.45\,;\; H = 31\%$$

Remark

- ✔ For a TTL $T = 900$ sec, hit rates are over 80% for all three traces
- ➔ High variability of inter-query times
  - – Burst arrivals in a short interval: rapidly increasing hit rates up to a certain TTL
  - – Heavy-tailed $F(t)$: diminishing marginal returns from increasing TTLs

Conclusion

- ✔ Formulated the cache hit rate based on a renewal assumption for the inter-query arrival times.
- ✔ Analyzing extensive DNS traces shows that out model predicts observed statistics remarkably well.
- ✔ On-going work
  - Extension to multi-level cache structure in which TTL is drawn from a certain distribution.
  - Inaccuracy of the renewal simplifying assumption.