CS294-1 On-line Computation & Network Algorithms **Spring 1997**

Lecture 16: March 20

Lecturer: Yair Bartal *Scribe: Felix Wu*

In this lecture, we describe and begin to analyze the Heat & Dump algorithm for distributed paging on a uniform network [ABF93b]

16.1 Heat & Dump Algorithm

The basic idea behind the Heat & Dump algorithm (HD) is that we can mostly use a uniprocessor paging stratgy to determine evictions, but we do not want to erase the last copy of a file, so we need some sort of global memory management strategy which might sometimes disrupt the local uniprocessor strategies.

16.1.1 Preliminary Definitions

Let $\mathcal{L}_p$ denote the local paging strategy used by processor p.

Definition 16.1 *The* virtual configuration of processor p at time t, denoted $V(p,t)$, is the configuration achieved by $\mathcal{L}_p$ at time t. That is, the virtual configuration is the one that p would be in if it were an isolated processor. The *real configuration, $R(p,t)$, is the actual configuration of processor p at time t.*

Every page in every processor is marked with one of three labels: HOT, COLD, or DUMPED, with such pages belonging to $H(p,t)$, $C(p,t)$, or $D(p,t)$, respectively. Clearly, $R(p,t) = H(p,t) \cup C(p,t) \cup D(p,t)$.

Definition 16.2 *A file F is* single if there is only one copy of F in the network, and *nonsingle otherwise.*

All files are initially single and marked DUMPED.

16.1.2 The Algorithm

For a read request (p,F) at time t:

1. Compute $V(p,t+1)$. (i.e. feed the request F to $\mathcal{L}_p$ to update the virtual configuration.)

2. Erase all nonsingle file copies in $R(p,t) - V(p,t+1)$ from p.
3. If all processors are filled to capacity, and there are no nonsingle file copies marked COLD,
   (a) (Pre-end of phase) Mark every COLD file requested during the phase as DUMPED.
   (b) (End of phase) Mark every HOT file COLD.
   (c) (Beginning of next phase) Continue with 4.
4. Mark all files in $R(p,t) - V(p,t+1)$ DUMPED.
   (a) If F is in p, mark it HOT and exit.
   (b) If p is filled to capacity, select an arbitrary DUMPED file B in p, copy B into a buffer and erase it.
   (c) Copy F into p and mark it HOT.
   (d) If F was marked DUMPED in the processor from which it was copied, erase F from that processor.
   (e) If the buffer is not empty,
      i. If every processor is filled to capacity, select uniformly at random a nonsingle COLD file copy (G,q), and erase G from q.
      ii. Copy B into any processor not filled to capacity. (This will be q if the previous step was executed.) Mark B DUMPED.

Let $c(p)$ be the competitive ratio of $\mathcal{L}_p$, and let $c = \max_p c(p)$. (If all processors use a marking algorithm for their local strategy, $c = O(\log k)$, where $k = \max_p k_p$.)

Theorem 16.3 *HD is $O(\max\{log(m-f), c\})$-competitive against an oblivious adversary.*

16.1.3 Further Definitions

We will need a number of definitions for the analysis.

Definition 16.4 *The* strongly real configuration $\overline{R}(p,t) = R(p,t) - D(p,t)$.

Definition 16.5 *F is* hallucinatory at time t if $F \in V(p,t) - \overline{R}(p,t)$. F is *continuously hallucinatory at time t if F is hallucinatory now, and has been so since the end of the previous phase. If F is hallucinatory, but not continuously so, it is* newly hallucinatory.

We further distinguish between two types of newly hallucinatory pages, *genuine ones and* artificial ones. Genuine pages are those which were nonsingle COLD pages erased to make room for a DUMPED page. Artificial pages are those which became DUMPED during the pre-end of the phase.

These labels also apply to requests, where a request is considered hallucinatory if the file being requested is.

We observe that if F is hallucinatory in p, then it must have been in $V(p,t)$ since the beginning of the phase. This follows from the fact that pages enter $V(p,t)$ only when requested and requested pages always remain HOT until the end of the phase. (HOT pages are in $\overline{R}(p,t)$ and hence are not hallucinatory.)

16.1.4 Analysis

Let $\overline{\sigma}$ denote the request sequence, let λ denote the local coin tosses (i.e. those made by $\mathcal{L}_p$), and let γ denote the global coin tosses.

Lemma 16.6 *For fixed γ, $E_\lambda(NON(\overline{\sigma})) \leq c \cdot ADV(\overline{\sigma})$, where $NON(\overline{\sigma})$ denotes the cost of serving nonhallucinatory requests in $\overline{\sigma}$.*

Proof: Nonhallucinatory files are either not in $V(p,t)$ or are in $\overline{R}(p,t)$. If the latter, HD incurs no cost. For those requests which are not in the virtual configuration, we can bound the cost by $\sum_p \mathcal{L}_p(\overline{\sigma}(p))$, which is in turn bounded by $c \cdot \sum_p \mathrm{ADV}_p(\overline{\sigma}(p))$, by the c-competitiveness of the local algorithms. Finally, we note that the sum of the costs of the local adversaries is bounded by the cost of the global adversary, since the local adversaries could simply mimic the behavior of the global adversary. This implies the lemma. ■

In what follows, we consider λ to be fixed.

Lemma 16.7 *The lengths of the phases are independent of γ.*

This lemma allows the proofs that follow to proceed by induction on the number of phases.

Let $\overline{\sigma} = \overline{\sigma}_0\sigma$, where $\overline{\sigma}_0$ is the request sequence up to the current phase, and σ is the request sequence for the current phase. Similarly, we can fix γ, and let $\gamma = \gamma_0\tau$.

Let t_0 denote the time at the end of the previous phase, let t_e^- denote the time at the pre-end of the current phase, and let t_e denote the time at the end of the current phase.

We will now want to define an amortized adversary cost in the current phase at time t, which we denote by $A(t)$. We can then define the accumulated amortized adversary cost as $\overline{A}(\overline{\sigma}) = \overline{A}(\overline{\sigma}_0) + A(t_e)$, setting $\overline{A}$ to be 0 on the empty sequence.

Lemma 16.8 *$ADV(\sigma) \geq \Omega(1) \cdot E_\gamma(\overline{A}(\sigma))$.*

Let $\mathrm{nh}(\tau, t)$ be the number of newly hallucinatory files at time t.

Lemma 16.9 *For $t \leq t_e^-$, $nh(\tau,t) \leq A(t)$. At $t = t_e$, $nh(\tau, t_e) \leq 2A(t_e)$.*

Using this lemma, we can now show the following.

Lemma 16.10 *$CH(\overline{\sigma}) \leq 2\overline{A}(\overline{\sigma})$, where $CH(\overline{\sigma})$ denotes the cost of serving continuously hallucinatory requests.*

Defining $\mathrm{NH}(\sigma)$ analogously for newly hallucinatory requests, we get the following lemma.

Lemma 16.11 *$E_\tau(NH(\sigma)) \leq O(1) \cdot E_\tau(NON(\sigma)) + A(t_e) + O(\log(m-f)) \cdot \max_{\tau;\, t \leq t_e^-}\{nh(\tau,t)\}$.*

Finally, we can put together the various types of requests to show that

$$\begin{aligned}
E_{\gamma,\lambda}(\mathrm{HD}(\overline{\sigma})) &= E_{\gamma,\lambda}(\mathrm{NON}(\overline{\sigma}) + \mathrm{CH}(\overline{\sigma}) + \mathrm{NH}(\overline{\sigma})) \\
&\leq O(1) \cdot E(\mathrm{NON}(\overline{\sigma})) + O(\log(m-f)) \cdot E(\overline{A}(\overline{\sigma})) \\
&\leq O(\max\{\log(m-f), c\}) \cdot \mathrm{ADV}(\overline{\sigma})
\end{aligned}$$

thus proving the stated competitiveness of Heat & Dump.

References

[ABF93b] B. Awerbuch, Y. Bartal, and A. Fiat, *Heat & Dump: Randomized competitive distributed paging*, in Proc. 34th IEEE Symp. on Foundations of Computer Science, IEEE, pages 22–31,November 1993.