Recitation 3

March 14, 2013

Problem

The *Hadamard matrices* $H_0, H_1, H_2, \ldots$ are defined as follows:

- H_0 is the 1×1 matrix $[1]$.
- For $k > 0$, H_k is the $2^k \times 2^k$ matrix

$$H_k = \left[\begin{array}{c|c} H_{k-1} & H_{k-1} \\ \hline H_{k-1} & -H_{k-1} \end{array}\right]$$

Throughout this problem assume that the numbers involved are small enough that basic arithmetic operations like addition and multiplication take unit time.

a. What is H_0? What is H_1? What is H_2?

b. Compute $H_0 \cdot (z)$, $H_1 \cdot \begin{bmatrix} x \\ y \end{bmatrix}$ and $H_2 \cdot \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$, where a, b, c, d, x, y and z are numbers.

c. Assume you have a black-box that computes $H_{k-1} \cdot v$ for any column vector v (of the appropriate size). Show how to use two calls to this black box, plus $O(2^k)$ additional work, to compute $H_k \cdot u$ for any column vector u (of the appropriate size). Briefly prove the correctness of your approach.

d. Design a recursive algorithm that takes as input a non-negative integer k and a column vector v of appropriate size and computes $H_k \cdot v$.

e. Analyze (and prove) your algorithm's runtime. Express the runtime in terms of the size of the input vector $n = 2^k$.

Solution

Part (a)

$$H_0 = \begin{bmatrix} 1 \end{bmatrix} \qquad H_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{1}$$

Part (b)

$$H_0 \cdot (z) = \begin{bmatrix} z \end{bmatrix} \qquad H_1 \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x+y \\ x-y \end{bmatrix} \qquad H_2 \cdot \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} a+b+c+d \\ a-b+c-d \\ a+b-c-d \\ a-b-c+d \end{bmatrix} \tag{2}$$

Part (c)

Let $\vec{v}_{top}$ be a vector of size 2^{k-1} consisting of the first half of $\vec{v}$, and let $\vec{v}_{bot}$ be a vector of the same size consisting of the second half of $\vec{v}$.
Call our black box on $\vec{v}_{top}$ and assign the output to a vector $\vec{a}$. Do the same for $\vec{v}_{bot}$ and assign the output to a vector $\vec{b}$.
Let $p = \vec{a} + \vec{b}$ and $q = \vec{a} - \vec{b}$ be vectors of size 2^{k-1}. We then form $\vec{v'} = H_k \cdot \vec{v}$ by appending $\vec{q}$ to $\vec{p}$.

Proof of Correctness

Consider the following derivation:

$$\begin{aligned}
H_k \cdot \vec{v} &= \begin{bmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{bmatrix} \cdot \begin{bmatrix} v_0 \\ v_1 \\ \dots \\ v_{2^k} \end{bmatrix} && \text{definition of } H_k \ (1) \\
&= \begin{bmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{bmatrix} \cdot \begin{bmatrix} \vec{v}_{top} \\ \vec{v}_{bot} \end{bmatrix} && \text{definition of } \vec{v}_{top} \text{ and } \vec{v}_{bot} \ (2) \\
&= \begin{bmatrix} [H_{k-1} \cdot \vec{v}_{top} + H_{k-1} \cdot \vec{v}_{bot}] \\ [H_{k-1} \cdot \vec{v}_{top} - H_{k-1} \cdot \vec{v}_{bot}] \end{bmatrix} && \text{matrix multiplication } (3) \\
&= \begin{bmatrix} \vec{a} + \vec{b} \\ \vec{a} - \vec{b} \end{bmatrix} && \text{definition of } a \text{ and } b \ (4) \\
&= \begin{bmatrix} \vec{p} \\ \vec{q} \end{bmatrix} && \text{definition of } p \text{ and } q \ (5) \\
&= \vec{v'} && \text{definition of } \vec{v'} \ (6)
\end{aligned}$$

The only step that may need a little more justification is (3).
To see why (3) is true, consider the process of computing the matrix multiplication in (2). To get the i^{th} element of the resulting vector, we compute the sum

$$\sum_{j=1}^{2^k} h_{ij} v_j$$

We can break this sum into these two chunks:

$$\left(\sum_{j=1}^{2^{k-1}} h_{ij} v_j \right) + \left(\sum_{j=1+2^{k-1}}^{2^k} h_{ij} v_j \right)$$

Now when $1 \le i \le 2^{k-1}$, we see that these two terms are in fact just

$$\left[i^{th} \text{ row of } H_{k-1} \right] \cdot \vec{v}_{top} + \left[i^{th} \text{ row of } H_{k-1} \right] \cdot \vec{v}_{bot}$$

And when $1 + 2^{k-1} \le i \le 2^k$, then the terms become

$$\begin{aligned}
&\left[i^{th} \text{ row of } H_{k-1} \right] \cdot \vec{v}_{top} + \left[i^{th} \text{ row of } -H_{k-1} \right] \cdot \vec{v}_{bot} \\
= &\left[i^{th} \text{ row of } H_{k-1} \right] \cdot \vec{v}_{top} - \left[i^{th} \text{ row of } H_{k-1} \right] \cdot \vec{v}_{bot}
\end{aligned}$$

Hence the first half of the entries of the resulting vector are given by the following vector sum:

$$[H_{k-1} \cdot \vec{v}_{top}] + [H_{k-1} \cdot \vec{v}_{bot}]$$

And the second half is given by

$$[H_{k-1} \cdot \vec{v}_{top}] - [H_{k-1} \cdot \vec{v}_{bot}]$$

as claimed in (3).

Part (d)

APPLY-HADAMARD$(k, \vec{v})$

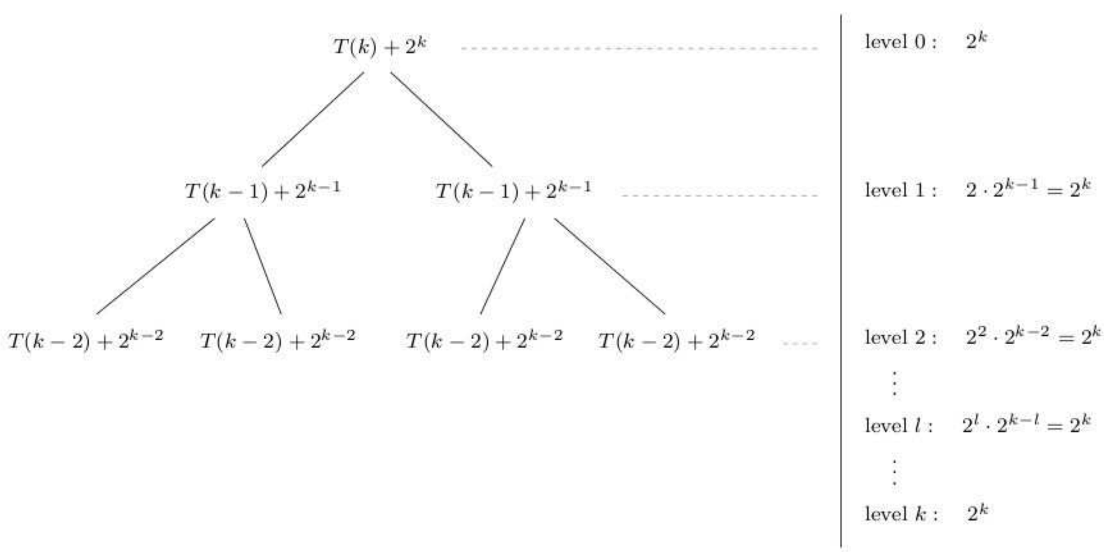
Takes a non-negative integer k and a vector $\vec{v}$ (of the appropriate size) and returns $H_k \cdot \vec{v}$.

if $k = 0$
    then return $\vec{v}$
else do
    $\vec{v}_{top} \leftarrow$ a vector consisting of the first 2^{k-1} components of $\vec{v}$
    $\vec{v}_{bot} \leftarrow$ a vector consisting of the last 2^{k-1} components of $\vec{v}$
    $\vec{a} \leftarrow$ APPLY-HADAMARD$(k-1, \vec{v}_{top})$
    $\vec{v} \leftarrow$ APPLY-HADAMARD$(k-1, \vec{v}_{bot})$
    $\vec{p} \leftarrow \vec{a} + \vec{b}$
    $\vec{q} \leftarrow \vec{a} - \vec{b}$
    $\vec{v'} \leftarrow \vec{q}$ appended to $\vec{b}$
    return $\vec{v'}$

Part (e)

The recursive algorithm given above makes two calls to itself on input $k-1$, and then does 2^k additional work when it does constant-time computations to find each entry of v'. So, the height of the recursive-call tree is k, since after k recursive calls $k = 0$ and the recursion ends. Referring to the diagram of the recurrence tree below (where T denotes the recursive function), we see that the total work done on a given level l of the tree is thus $2^l \cdot 2^{k-l} = 2^k$, where the level of the root is 0. Hence the total work done over all levels of the tree is $k \cdot 2^k$, giving a time complexity of $O(k \cdot 2^k)$ for our algorithm.

Recurrence Tree for APPLY-HADAMARD(k, v)



$\square$