Solutions to the Problems of Chapter 8

Petar Popovski

1. Let T_n denote the average number of slots required to complete binary tree algorithm when there are initially n tags. We will derive a recursive relation that relates T_n so the values T_k where $k < 1$. The initial conditions are $T_0 = 1$ (only one idle slot when there are no tags) and $T_1 = 1$ (one single-response slot when there is a single tag). The initial conditions are identical for basic binary tree protocol and the modified binary tree protocol. In deriving these equations it is helpful to think in terms of the framework with enabled intervals.

(a) We first derive T_2:

$$T_2 = 1 + \frac{1}{4}(T_2 + T_0) + \frac{1}{2}(T_1 + T_1) + \frac{1}{4}(T_0 + T_2) \tag{8.1}$$

where:

- The first summand 1 is due to the collision that always occurs when starting the tree protocol with $n = 2$ tags.
- The second summand denotes that, with probability $\frac{1}{4}$ the tokens of the two tags will be in the interval $[0, 0.5)$. Thus, the resolution of the interval $[0, 0.5)$ will take on average T_2 slots and we have to add to it the slot $T_0 = 1$ used when the reader enables the interval $[0.5, 1)$ that does not contain any token.
- The third summand denotes that, with probability $\frac{1}{2}$, there will be one token in $[0, 0.5)$ and one in $[0.5, 1)$. In that case, in addition to the initial collision, there will be $T_1 + T_1 = 2$ additional slots.
- The last summand accounts for the case in which there are no tokens in $[0, 0.5)$ and two tokens in $[0.5, 1)$

If (8.1) is solved using only T_2 as unknown, we obtain:

$$T_2 = 5 \tag{8.2}$$

Using the reasoning above, the expression for T_3 can be derived as:

$$T_3 = 1 + \frac{1}{8}(T_3 + T_0) + \frac{3}{8}(T_2 + T_1) + \frac{3}{8}(T_1 + T_2) + \frac{1}{8}(T_0 + T_3) \tag{8.3}$$

Using the value of T_2 from (8.2), the value of T_3 is found to be:

$$T_3 = \frac{23}{3} \tag{8.4}$$

The expressions for general value of n can be found in:

R. Rom and M. Sidi. *Multiple Access Protocols: Performance and Analysis.* Springer-Verlag, 1991.

(b) In the modified binary tree algorithm, if an idle slot follows after a collision, then the next certain collision is avoided. Now the expression for T_2 is:

$$T_2 = 1 + \frac{1}{4}(T_2 + T_0) + \frac{1}{2}(T_1 + T_1) + \frac{1}{4}(T_0 + T_2 - 1) \tag{8.5}$$

where -1 within the last summand denotes that one slot is saved by avoiding certain collision in $[0.5, 1)$ when there are no tokens discovered in $[0, 0.5)$. Solving this equation gives $T_2 = 4.5$.

The expression for T_3 when MBT is used is:

$$T_3 = 1 + \frac{1}{8}(T_3 + T_0) + \frac{3}{8}(T_2 + T_1) + \frac{3}{8}(T_1 + T_2) + \frac{1}{8}(T_0 + T_3 - 1) \tag{8.6}$$

which results in $T_3 = 7$.

(c) If there are $n = 2$ tags and the reader knows that in advance, then

$$T_2 = 1 + \frac{1}{4}T_2 + \frac{1}{2}T_1 + \frac{1}{4}T_2 \tag{8.7}$$

Now the summand 1 has a different interpretation. It is the slot that is spent when initially the interval $[0, 0.5)$ is enabled. If, with probability $\frac{1}{4}$, there is collision in this first slot, then the average remaining duration is still T_2, as the reader knows that there are two tokens in $[0, 0.5)$, which is equivalent to the knowledge it had at the beginning (two tokens in $[0, 1)$. However, after $[0, 0.5)$ is resolved and two tokens are discovered, the reader will not enable $[0.5, 1)$ as it knows there are no tokens. Continuing to analyze the sum in (8.7), with probability $\frac{1}{2}$ the interval $[0, 0.5)$ contains one token such that only T_1 slots will be used in resolving $[0.5, 1)$. The reader can analogously find out the meaning of the third summand. Solving (8.7), we obtain $T_2 = 3$.

When $n = 3$, the first enabled interval is $[0, \frac{1}{3})$. Then the following can happen:

- If the reader observes collision in this interval, then it does not know whether there are three or two tokens in $[0, \frac{1}{3})$. Therefore, after observing that collision, a good move is to enable the interval $[\frac{1}{3}, 1)$. If that interval is empty, then it knows that there are three tokens in $[0, \frac{1}{3})$. On the other hand, if there is a single token in $[\frac{1}{3}, 1)$, then it knows that there are two tokens in $[0, \frac{1}{3})$. Note that there can be no collision in $[0, \frac{1}{3})$ and $[\frac{1}{3}, 1)$, as there are only three nodes.
- If the reader observes a single reply in $[0, \frac{1}{3})$, then it knows there are two tokens in $[\frac{1}{3}, 1)$.
- If the reader observes no tokens in $[0, \frac{1}{3})$ (idle slot), then it knows there are three tokens in $[\frac{1}{3}, 1)$.

Putting the above statements in formulas, we obtain:

$$T_3 = 1 + \left(\frac{1}{3}\right)^3 (T_0 + T_3) + \binom{3}{2}\left(\frac{1}{3}\right)^2 \frac{2}{3}(T_1 + T_2) + \binom{3}{1}\frac{1}{3}\left(\frac{2}{3}\right)^2 T_2 + \left(\frac{2}{3}\right)^3 T_3 \tag{8.8}$$

where the second summand contains T_0 due to the idle slot perceived when $[\frac{1}{3}, 1)$ is enabled. The third summand contains T_1 due to the single slot perceived when $[\frac{1}{3}, 1)$ is enabled. Solving (8.8) and using the derived value $T_2 = 3$, we find $T_3 = \frac{44}{9}$.

2. The pseudocode is given below. This version of the tree protocol is referred to as stack algorithm, originally invented by B. Tsybakov.

```
$level = 0$; state= $active$;
if initiation signal received
    transmit;
while state== $active$
    receive $feedback$;
    if $feedback$==COLLISION
        if $level == 0$
            $b = randombit$;
            if $b == 1$
                $level = level + 1$;
    else
        if $feedback$==SINGLE
            if $level == 0$
                state= $terminated$;
            else
                $level = level - 1$;
        else
            $level = level - 1$;
    if $level == 0$
        transmit;
```

3. The problem is that in this case the reader does not have a deterministic criterion to decide whether the collision has been resolved. After the reader initiates the tree protocol and it does not observe the channel in a single state, then it cannot know whether that is because there are no tags or there are $n > 1$ tags. Hence, in this case the reader should preferably treat the slots in which it does not perceive successful transmission as collisions rather than idle. However, it should not deterministically treat each of the no-success slots as a collision – this is because if the real state of the slot is idle, then the protocol enters into an infinite loop. Algorithms to treat these cases have been described in:

T. Berger, N. Mehravari, D. Towsley, and J. K. Wolf, "Random multiple-access communication and group testing," *IEEE Trans. Commun.*, vol. COM-32, no. 7, pp. 769–779, July 1984.

4. If the framed ALOHA has K slots, then the probe sent by the reader can be understood as enabling the following intervals in a sequential manner: $[0, \frac{1}{K})$, $[\frac{1}{K}, \frac{2}{K})$, $\ldots [\frac{K-1}{K}, 1)$. The tags generate random tokens. Each tag transmits in the slot in which the enabled interval contains its token. At the end of the frame, the tags receive feedback about the slots in which a single transmission was observed by the reader. Then the reader sends another probe and initiates another frame, possibly with a different length K_1. The unresolved tags generate new random tokens. Note that this is different from tree protocols, where the tokens are retained. This explains the fundamental difference between tree protocols and ALOHA: tree protocols persistently resolve collisions, while ALOHA protocols randomize the transmissions in order to avoid collisions.

5. After a clipped binary tree (CBT) is terminated, the EBT algorithm enables an interval of length which 2^{-L}, where L is selected such that, based on an estimate $\hat{n}$ of the total tag population, the probability of observing a single transmission in that interval is maximized. This is achieved by making $2^L \approx \hat{n}$. Recall that a probe in Class 1 protocol enables, in a sequential manner, eight intervals. Therefore, the bit mask used in PingID should contain L bits such that $2^{L+3} \approx \hat{n}$.

6. This is because the collision resolution algorithm will continue to resolve the "collision". If an idle slot is interpreted as collision, then, if there are no errors in the next two slots, they will both be perceived as an idle. On the other hand, if a single is interpreted as a collision, and there are no errors in the next two slots, then one will be perceived as collision and the other as single.