Appendix for "Minimizing the AoI for Pull-Based Target-Level Data Collection in Energy-Harvesting IoTs"

Bingkun Yao, Hong Gao, Yang Zhang, Dongjing Miao, Quan Chen and Jianzhong Li

APPENDIX

A. PROOF OF THEOREM 1

We prove Theorem 1 by a special case of AoI-Pull-Target, which is described as follows:

1) The length of the whole monitoring duration is 2 timeslots (i.e., K=2).

2) For each target t and i = 1 or 2, there exists at least one query q = (t_q, [s_q, e_q]) $\in Q$ such that $t_q = t$, $s_q = e_q = i$.

3) For each node $n \in N$, $B_n(1) = E_c(n)$, $EH_n(1) = EH_n(2) < E_c(n)/2$.

4) $|N| \leq M$.

Now we show that a decision problem of the above special case is NP-Complete. That is, whether there exists a working schedule $Sdl(n, i)$ for each $n \in N$ and $i \in \{1, 2\}$, so that AVG_AoI = 0? Obviously, in case that AVG_AoI = 0, we have $A(t, i)$ = 0 for each $t \in T$ and vice versa. According to (2) above, for each $t \in T$ and $i \in \{1, 2\}$, we have $|Q(t, i)| > 0$. Under this condition, according to the definition of $A(t, i)$, if $A(t, i)$ = 0, i is equal to $U(t, i)$. By the definition of $U(t, i)$, this means that for each $t \in T$ and $i \in \{1, 2\}$, the sensed data of t is sampled and transmitted by at least one node that can cover t. In other words, let $SD(i)$ denote the set of nodes that are scheduled to be active at timeslot i, then we have $t \in \underset{n \in SD(i)}{\cup} \Gamma(n)$. According to (3) and (4) above, in the whole monitoring duration, each node can choose to be active at timeslot 1 *or* 2. Therefore, the above decision problem is transformed into another problem: Whether we can divide the node set N into two disjoint sets $SD(1)$ and $SD(2)$ such that for each $t \in T$ and $i \in \{1, 2\}$, $t \in \underset{n \in SD(i)}{\cup} \Gamma(n)$? This problem is known as 2-DSC, which has been proved to be NP-Complete in [1]. This indicates that the above special case of AoI-Pull-Target is NP-Hard. Therefore, the problem of AoI-Pull-Target is NP-Hard.

B. PROOF OF THEOREM 2

We prove Theorem 2 by constructing a new solution Sdl' from Sdl^*, in which for $has\_l(n, i)+1 \leq j \leq ini\_active(n, i)-1$, if $Sdl^*(n, j) = 1$, $Sdl'(n, j) = 0$. Now we prove that Sdl' is also an optimal solution. Firstly, according to the definition of $SUM(n)$, since $|Q(t, j)| = 0$ for $has\_l(n, i)+1 \leq j \leq ini\_active(n, i)-1$ and n is guaranteed to be active at slot $ini\_active(n, i)$, it can be easily seen that $SUM'(n) = SUM^*(n)$. Also, as the active slots determined by Sdl' is a subset of those determined by Sdl^*, Sdl' also satisfies the energy and bandwidth constraint.

Manuscript received April 19, 2005; revised August 26, 2015.

C. PROOF OF THEOREM 3

We prove Theorem 3 by contradiction. Assume that there exists a working schedule $Sdl^+(n, k)$ for $i \leq k \leq NL(n)$ that satisfies $SU^+(i, l, e) < SU^*(i, l, e)$.

Firstly, we discuss the case when $i \in First\_active(n)$. As n is scheduled to be active at slot i by Phase 1, according to Eq. 16, we have:

$$SU^*(i, l, e) = SU^*(i+1, 1, B_n{}'(e, i, 1)) \tag{A1}$$

As $Sdl^+(n, k)$ satisfies Condition B2 defined in the subproblem of n, $Sdl^+(n, k)$ can be executed by node n from slot i+1 to $NL(n)$, which satisfies:

$$SU^+(i+1, 1, B_n{}'(e, i, 1)) = SU^+(i, l, e) < SU^*(i, l, e) \tag{A2}$$

By combining Eq. A1 and A2, we have $SU^+(i+1, 1, B_n{}'(e, i, 1)) < SU^*(i+1, 1, B_n{}'(e, i, 1))$, which conflicts with the definition that $SU^*(\cdot)$.

Secondly, when $i \notin First\_active(n)$, there are two cases:

1) $e < E_c(n)$ or $AC(i) + 1 > M$. Under this condition, n can only be idle in slot i. This scenario can be proved similarly to the case when $i \in First\_active(n)$.

2) $e \geq E_c(n)$ and $AC(i) + 1 \leq M$. If $Sdl^+(n, i) = 0$, at timeslot i+1, we have $i+1-L_1(n, i+1) = l+1$ and $B_n(i+1) = B_n{}'(e, i, 0)$. If working schedule $Sdl^+(n, k)$ is applied on node n from timeslot i+1 to $NL(n)$, we have:

$$SU^+(i, l, e) - C(n, i, l) \geq SU^*(i+1, l+1, B_n{}'(e, i, 0)) \tag{A3}$$

Mention that Eq. A3 holds by the definition of $SU^*(\cdot)$. Also, as $SU^+(i, l, e) < SU^*(i, l, e)$, we have:

$$SU^*(i, l, e) - C(n, i, l) > SU^+(i, l, e) - C(n, i, l) \tag{A4}$$

By summarizing Eq. A3 and A4, we have $SU^*(i, l, e) > SU^*(i+1, l+1, B_n{}'(e, i, 0)) + C(n, i, l)$, which conflicts with Eq. 16. The case when $Sdl^+(n, i) = 1$ can be proved similarly.

D. Proof of Theorem 4

We prove Theorem 4 this by mathematics induction. Firstly, we consider the case when $i = NL(n)$. By Eq. 15, if n could be scheduled to be active at slot $NL(n)$, for each l, we have $SU^*(NL(n), l, e) = 0$. Otherwise, n stays idle at slot $NL(n)$. Since $l \geq Max\_A(n)$, according to Eq. 13 and 14, we have:

$$SU^*(NL(n), l, e) = \sum_{t \in \Gamma(n)} |Q(t, NL(n))| \left(NL(n) - U(t, NL(n))\right) \tag{A5}$$

which is a fixed value independent of l. Therefore, the theorem holds when $i = NL(n)$.

Next, given that the theorem is correct when $i = q < NL(n)$, we prove that the case when $i = q$-1 still holds true:

1) If $q\text{-}1 \in First\_active(n)$, According to Eq. 14, $SU^*(q\text{-}1, l, e) = SU^*(q, 1, e)$, which is a fixed value independent of l.

2) If $e < E_c(n)$ or $OB(q\text{-}1) + R(n) > M$, n can only choose to be idle and we have:

$$SU^*(q-1, l, e) = SU^*(q, l+1, e) + C(n, q-1, l) \tag{A6}$$

in which $C(n, q\text{-}1, l)$ is a fixed value. Also, since the case when $i = q$ holds true, for any $l_1, l_2 \geq Max\_A(n)$, $SU^*(q, l_1+1, e) = SU^*(q, l_2+1, e)$. Hence, $SU^*(q\text{-}1, l_1, e) = SU^*(q\text{-}1, l_2, e)$.

3) Otherwise, n can choose to be active or idle at slot q-1. Given l_1, $l_2 \geq Max\_A(n)$, by Eq. 14, this case can be considered under the following three conditions:

3-1) Both $SU^*(q\text{-}1, l_1, e)$ and $SU^*(q\text{-}1, l_2, e)$ are equal to $SU^*(q, 1, B_n'(e, q\text{-}1, 1))$. It is very obvious that $SU^*(q\text{-}1, l_1, e) = SU^*(q\text{-}1, l_2, e)$.

3-2) $SU^*(q\text{-}1, l_1, e) = SU^*(q, l_1+1, B_n'(e, q\text{-}1, 0)) + C(n, q\text{-}1, l_1)$ and $SU^*(q\text{-}1, l_2, e) = SU^*(q, l_2+1, B_n'(e, q\text{-}1, 0)) + C(n, q\text{-}1, l_2)$. This case can be verified similarly to case (b).

3-3) Otherwise, without loss of generality, assume that $SU^*(q\text{-}1, l_1, e) = SU^*(q, 1, B_n'(e, q\text{-}1, 1))$ and $SU^*(q\text{-}1, l_2, e) = SU^*(q, l_2+1, B_n'(e, q\text{-}1, 0)) + C(n, q\text{-}1, l_2)$. According to Equation (10), we have:

$$SU^*(q, l_2+1, B_n'(e, q-1, 0)) + C(n, q-1, l_2) \leq \tag{A7.1}$$

$$SU^*(q, 1, B_n'(e, q-1, 1)) \leq \tag{A7.2}$$

$$SU^*(q, l_1+1, B_n'(e, q-1, 0)) + C(n, q-1, l_1) \tag{A7.3}$$

Similar to case (b), we have A7.1 = A7.3. Thus, A7.1 = A7.2, which indicates that $SU^*(q\text{-}1, l_2, e) = SU^*(q\text{-}1, l_1, e)$.

In summary, the theorem remains valid when $i = q$-1. This ends the proof.

E. Proof of Theorem 5

Lemma 1. *Let Max_A denote the maximum value of i-$U(t, i)$ for all $t \in T$ and $1 \leq i \leq K$ satisfying that $|Q(t,i)| > 0$. After Phase 1, Max_A is upper bounded by $Max\_C + \frac{ln(|T|)C\_num}{M}$-1, in which Max_C is the maximum number of idle slots needed for a node $n \in N$ to harvest $E_c(n)$ units of energy and C_num is minimum number of nodes required to cover all the targets in T.*

Proof. Firstly, consider a node $n \in First\_active$ and during interval $[s_1, d_1]$, n is scheduled to be active at slot $ini\_active(n, 1)$. For each target $t \in \Gamma(n)$ and slot $j \in [1, ini\_active(n, 1)]$, consider the upper bound of the following metric denoted by $Max\_A(t, 1)$:

$$Max\_A(t, 1) = Max\left\{j - U(t, j) \,|\, |Q(t, j)| \neq 0\right\} \tag{A8}$$

According to Algorithm 1, if $ini\_active(n, 1) = Can\_ac(n)$, $Max\_A(t, 1) \leq Can\_ac(n)$-1. Also, according to Substep 2-1 of Phase 1 and Line 6 of Algorithm 1, we have $Can\_ac(n)$-1 $\leq Max\_C + \frac{|First\_Sdl|}{M}$-1. Let C_num denote the minimum number of nodes required to cover all the targets. Since $First\_Sdl$ is computed using the greedy heuristic for set cover, according to [2], $|First\_Sdl| \leq ln(|T|)C\_num$. Hence, we have:

$$Max\_A(t, 1) \leq Max\_C + \frac{ln(|T|)C\_num}{M} - 1 \tag{A9}$$

Otherwise, if $ini\_active(n, 1) = has\_q(n)$, for each slot $j \in [Can\_ac(n), has\_q(n)\text{-}1]$, $|Q(t,j)| = 0$. This yields the same upper bound for $Max\_A(t, 1)$ as Eq. A9. Therefore, for each $t \in T$ and $j \in [1, ini\_active(n, 1)]$ such that $|Q(t,j)| > 0$, j-$U(t, j)$ is upper bounded by Eq. A9. For each $t \in T$ and $j \in [ini\_active(n, i\text{-}1), ini\_active(n, i)]$ $(i \geq 2)$, it could be verified similarly that $Max\_A(t, i) \leq Max\left\{\frac{ln(|T|)C\_num}{M}, Max\_C\right\}$-1. To summarize, $Max\_A(t, i)$ is upper bounded by Eq. A9. □

Proof of Theorem 5. The proposed algorithm consists of Phase 1 and 2. Firstly, consider the time complexity of Phase 1 (Algorithm 1). The time complexity of Line 1 is $O(|N||T|)$. Line 2-3 consumes $O(K)$-time. From Line 4 to 11, when computing $Can\_ac(n)$ and $has\_q(n)$ for each $n \in First\_Sdl$, as each slot is considered only once, Line 4-12 consumes $O(|N|K)$-time. To summarize, the time complexity of Phase 1 is $O(|N|(|T|+K))$.

Next, we discuss the Phase 2 (Algorithm 3). Firstly, consider the time consumption of Algorithm 2: Line 5 of Algorithm 2 requires $O(AVG\_\Gamma)$ time, in which AVG_Γ is the average $|\Gamma(n)|$ for $n \in N$. Line 6-17 of Algorithm 2 consumes $O(B_{max})$-time. Line 5 to 17 of Algorithm 3 is repeated for $Min\{UB(n,i), Max\_A(n)\}$ times. According to Theorem 4 and Lemma 1, Since $Max\_A(n) \leq Max\_A$ and Line 3-17 is repeated for $O(K)$ times, the time complexity of Line 2 to 17 is $O[K(Max\_C + \frac{ln(|T|)C\_num}{M})(AVG\_\Gamma + B_{max})]$. The time cost of Line 18-21 and Line 22-24 is $O(K)$ and $O(K \times AVG(|\Gamma|))$ respectively. Then, as Algorithm 2 is repeated for $|N|$ times, the time complexity of Phase 2 is $O[|N|K(Max\_C + \frac{ln(|T|)C\_num}{M})(AVG\_\Gamma + B_{max})]$. Summing up Phase 1 and 2 ends the proof.

References

[1] M. Cardei and DZ. Du, "Improving wireless sensor network lifetime through power aware organization," in *Wireless networks*, vol. 11, pp. 333–340, 2005.

[2] V. Vazirani, "Approximation algorithms," Berlin: springer, 2001.