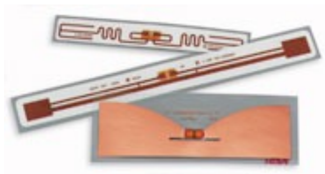
Autonomous Networking

Gaia Maselli
Dept. of Computer Science

Today's plan

- RFID system
  - Components, characteristics
- MAC protocols for RFID systems
  - Tree based
  - Aloha based

What is an RFID system?



RF Tags

Interrogators
and Antennas

Server
& Data repositories

Radio frequency labels store a unique identifier (ex. 96 bits) and consist of an antenna integrated on a microchip.

They are attached to object to be identified

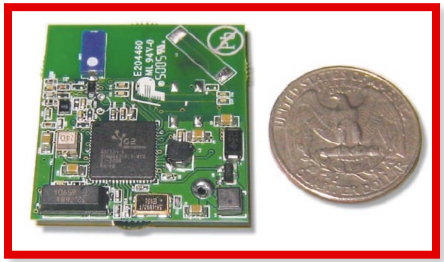
The reader queries tags to get their IDs

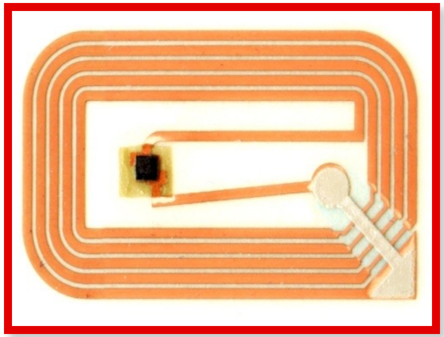
A server handles the data received by the reader and process it based on the application requeirements.

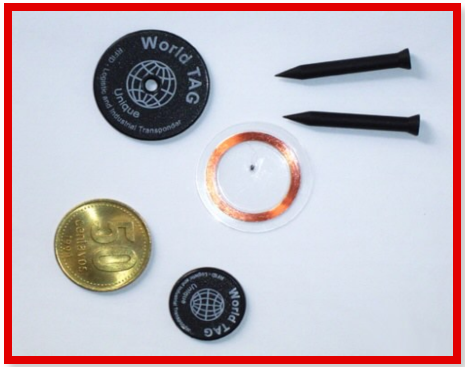
Main components

Tags

- Small, cheap, long lasting



Reader

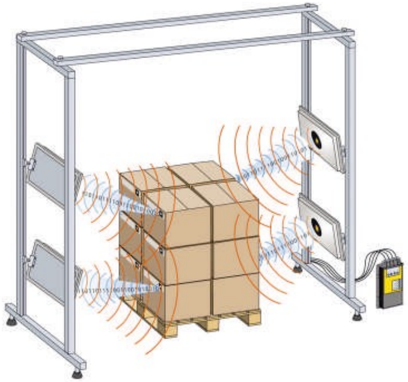
- powerful device

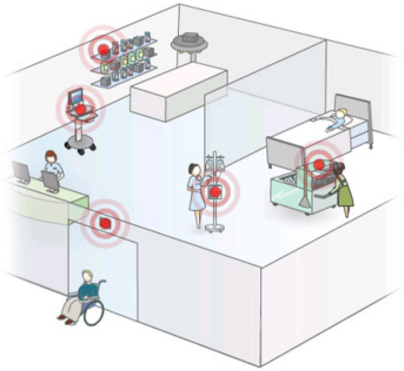

Application goal

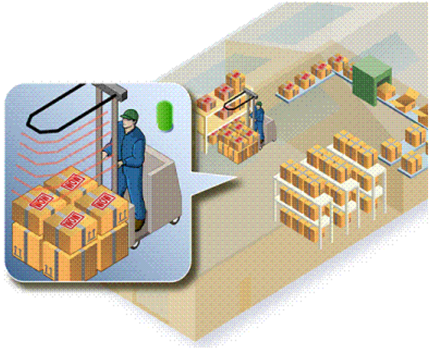
A variety of applications whose common required functionality is **object identification** — to get the unique ID associated to each tag (each object has a tag attached to it).

Applications

- Inventory and logistics
- Acces controll object tracking
  - Libraries
  - Airport luggages
- Domotics e Assisted Living
  - Intelligent appliances
  - Daily assistance to people with disabilities





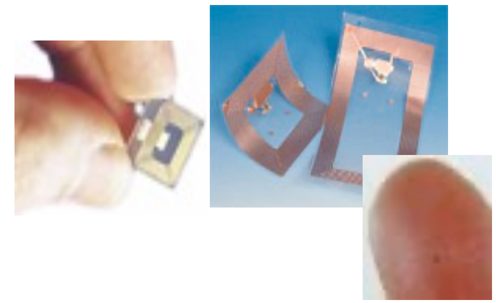


RFID: communication

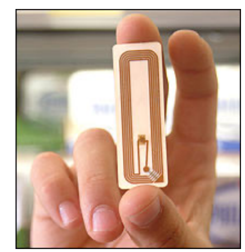
- Wireless communication
  - Reader to tags
  - Tags to reader
  - Tag ~~to tag~~

WHY ~~tag-to-tag~~ ?

- Passive tags



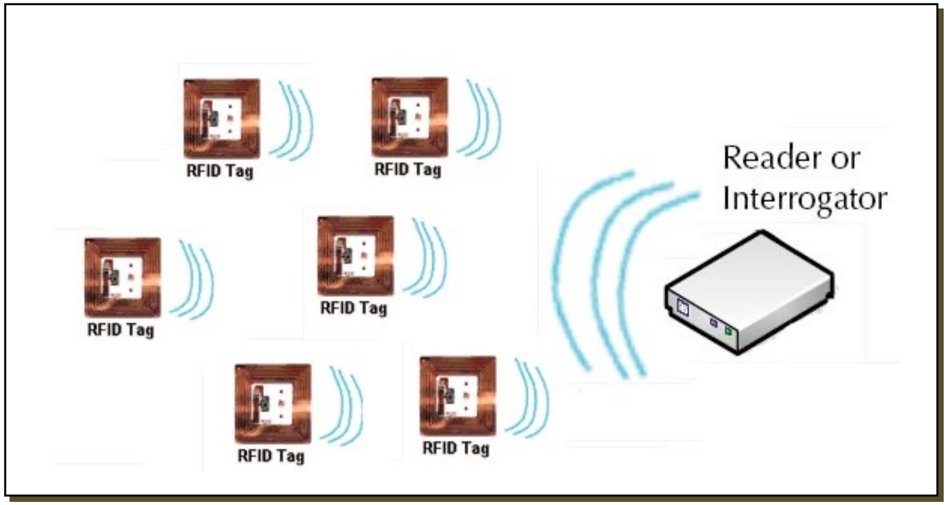
- No power source (no battery!)



- Transmission through **back-scattering**

RFID system

- **RFID** is the traditional and most widely used technology that **harvests power from RF signals**.
- In RFID, the **tags** — battery free devices — **reflect the high-power constant signal generated by the reader** — a powered device — to send it their unique ID.

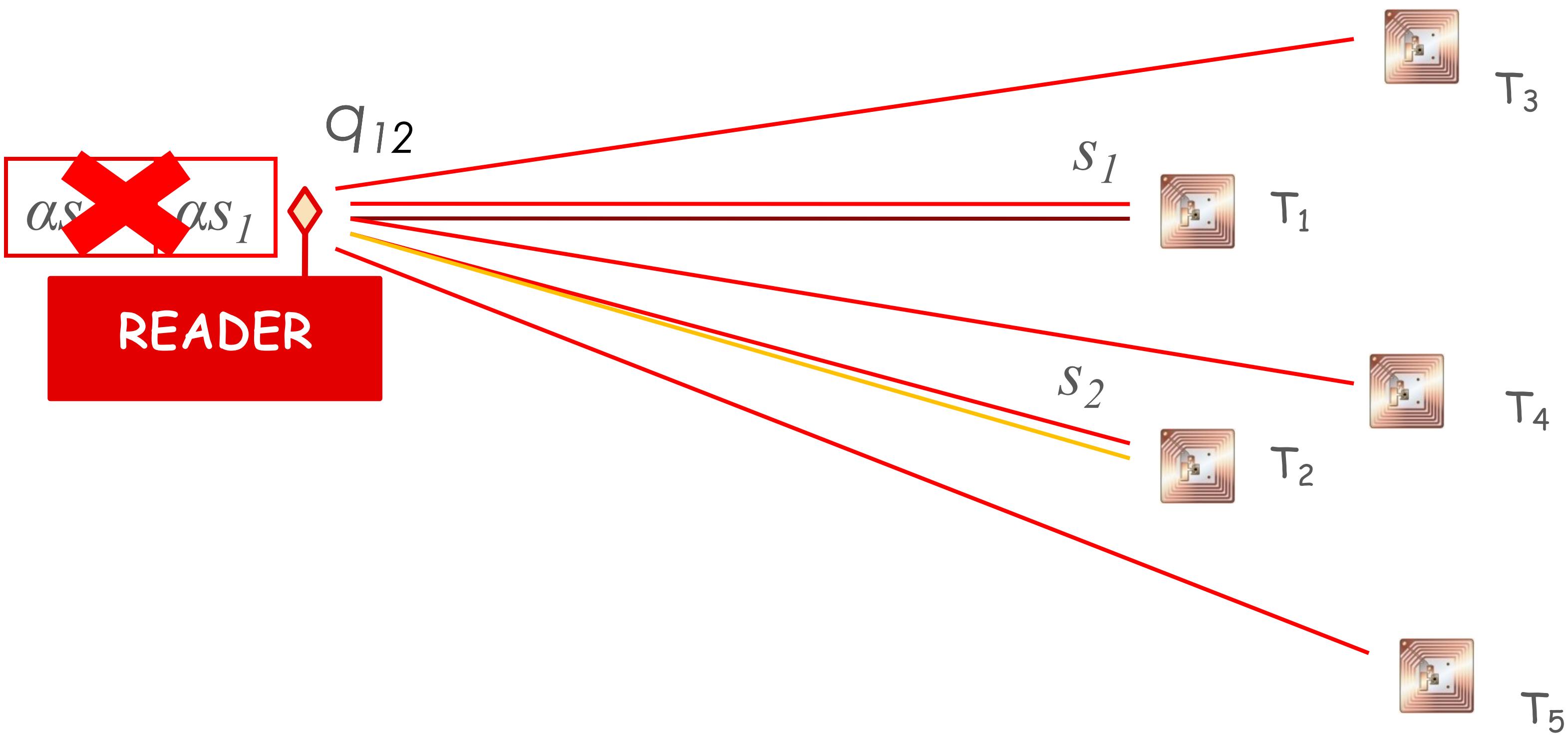


RFID channel

- Wireless
- Shared
- Low datarate (typically tag-to-reader is 40 kbps)
- If multiple tags reply they do it simultaneously (collision)

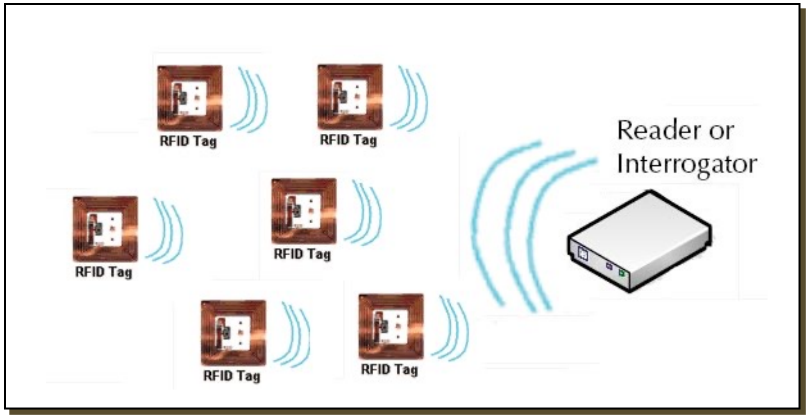
Tag reading

- RFID tags transmit their unique ID (typically 96 bits, maximum 256 bits)



MAC issues

- Large number of **passive** tags
- Tags **cannot transmit spontaneously**
  - Reader queries tags
  - Tags respond with their ID by back-scattering the received signal



- Simultaneous tag responses cause collision
- Tags cannot ear each other (**NO Carrier Sense, NO Collision Detection**)
- **Channel access must be arbitrated by the reader**

MAC protocols

Several MAC protocols have been proposed to identify tags in a RFID system

- **Sequential** protocols (aim at singulating tag transmissions)
  - Tree based
    - Binary Splitting
    - Query Tree
    - Variations (Query Tree Improved)
  - Aloha based
    - Framed Slotted Aloha
    - EPC Gen Standard
    - Tree Slotted Aloha
    - Variations (BSTSA)
- **Concurrent** protocols (exploit tags collisions) – not covered in this course

MAC protocols

Several MAC protocols have been proposed to identify tags in a RFID system

- **Sequential** protocols (aim at singulating tag transmissions)
  - Tree based
    - Binary Splitting
    - Query Tree
    - Variations (Query Tree Improved)
  - Aloha based
    - Framed Slotted Aloha
    - EPC Gen Standard
    - Tree Slotted Aloha
    - Variations (BSTSA)

Binary Splitting protocol

Binary splitting principle

BS recursively splits answering tags into two subgroups until obtaining single-tag groups.

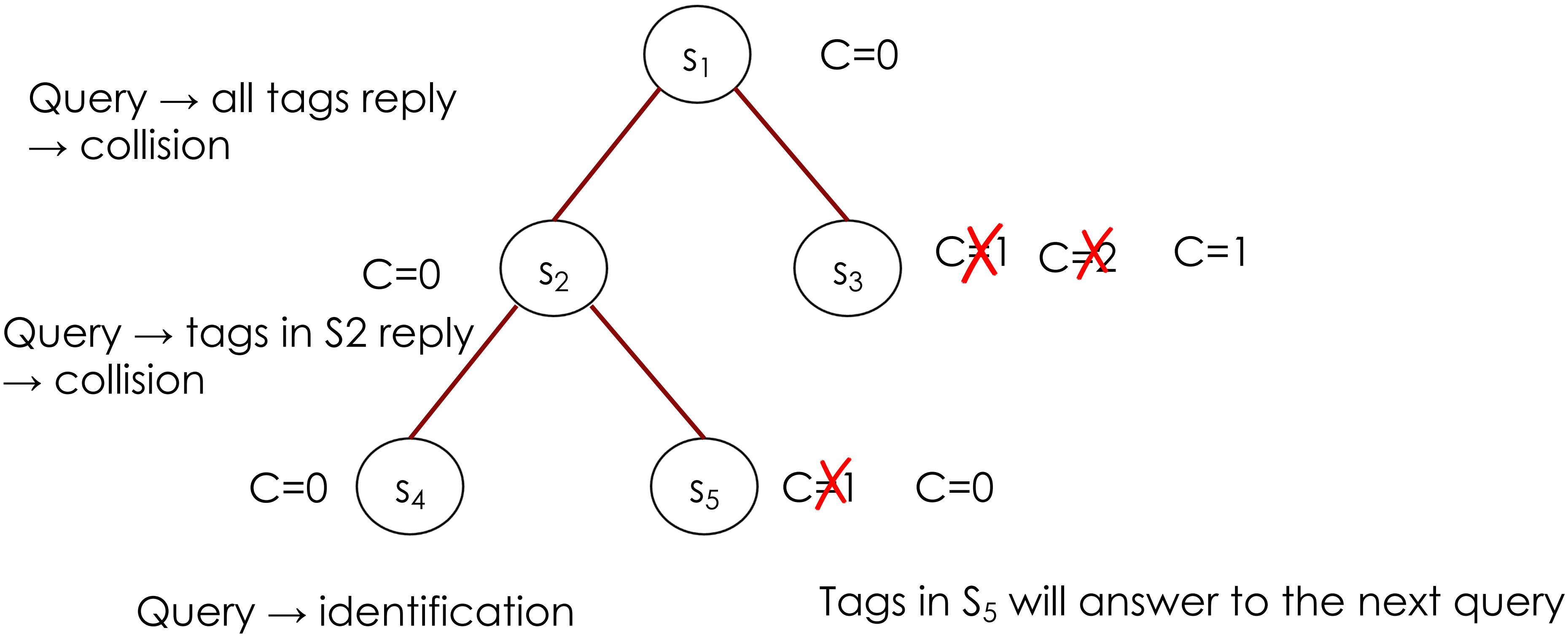
Tags answer to reader's queries according to the generation of a binary random number

Binary splitting

- Suppose we have a set of tags to identify
- Each tag has a counter initially set to zero.
- The tags with the counter = 0 reply to the reader query
- The reader sends a query
- All tags reply → collision
  - Each tag generates a random binary number (0,1) and sums it to the counter
- The process repeats
  - The reader sends a query
  - All tags with C=0 replies
  - If collision → each replying tag generates a random binary number and sums it to its counter
  - Each other tag (silent) → C=C+1
  - If none or one tag replies → all tags: counter = counter - 1

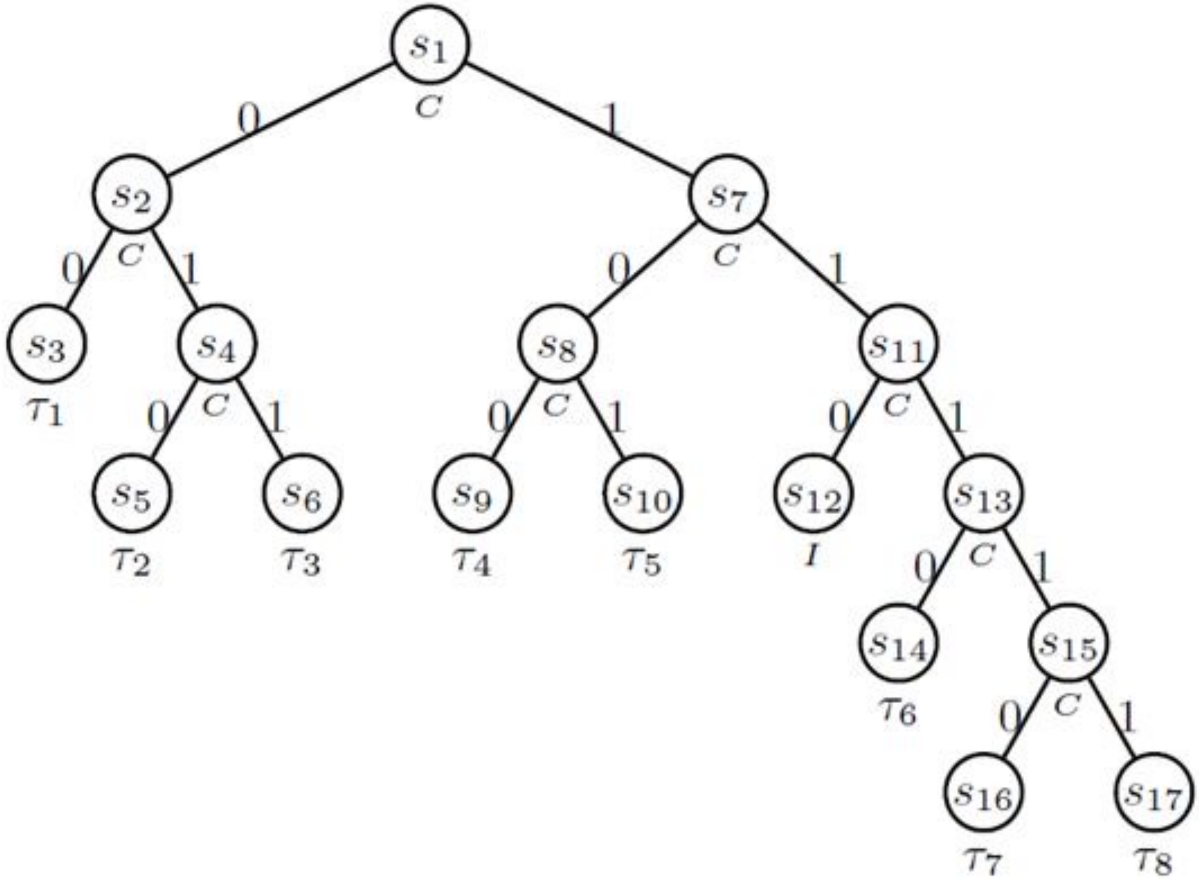
Binary splitting operation

- Suppose we have a set s_1 of 8 tags to identify



Binary Splitting: example

- Suppose we have a set s_1 of 8 tags to identify



Query Tree protocol

Query tree principle

QT queries tags according to the binary structure of their ID.

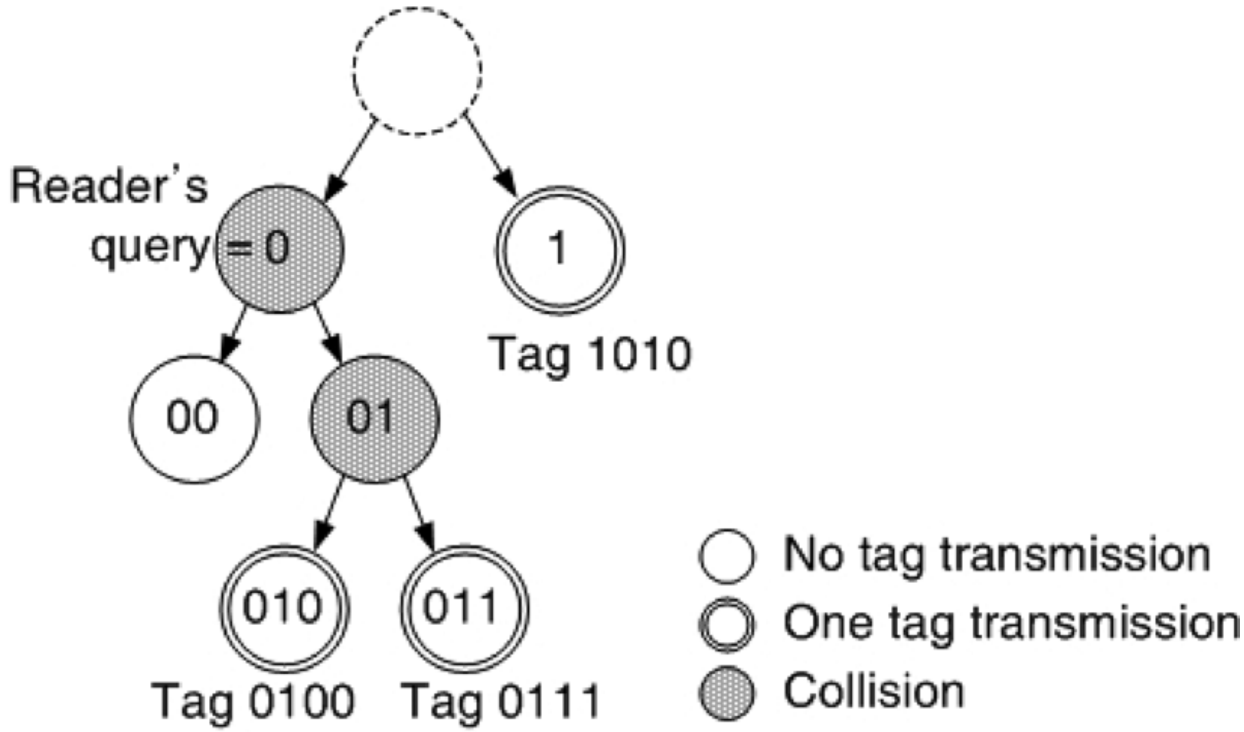
Each tag has an ID of typically 96 bits (but can be up to 256 bits long)

Query tree protocol

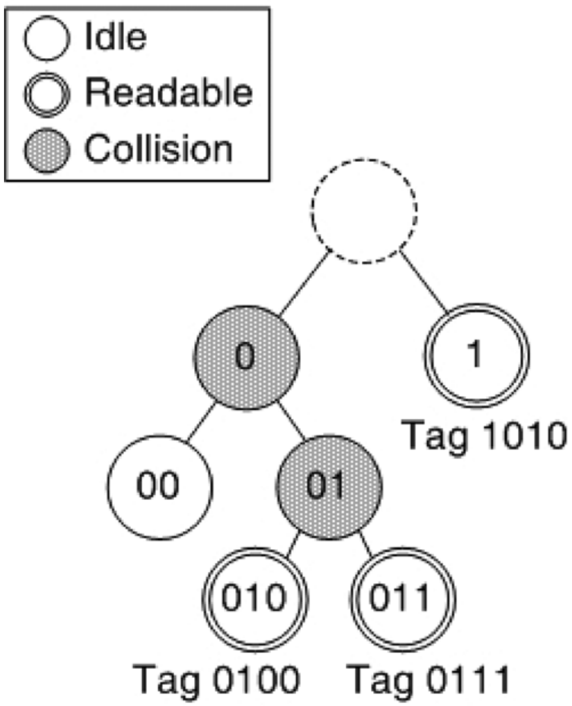
- The reader interrogates tags by sending them a **string**, and only **those tags whose IDs have a prefix matching that string respond to the query**.
- At the beginning, the reader queries all tags: this is implemented by including a **NULL** string in the query.
- If a **collision** occurs, then the **string length is increased by one bit** until the collision is solved and a tag is identified.
- The reader then starts a new query with a different string. In particular, if tag identification occurs with a string q0 the reader will query for string q1.

Query tree: example

- Suppose we have 3 tags whose IDs are:
- 0100
- 0111
- 1010



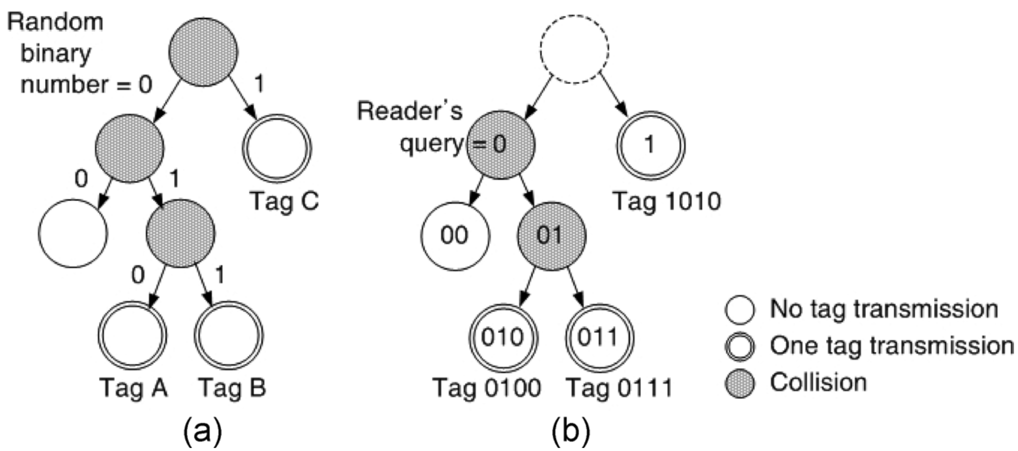
Query tree example



Step	Reader query	Tag responses: Tag 0100	Tag 0111	Tag 1010
0				
1	0	0100	0111	
2	1			1010
3	00			
4	01	0100	0111	
5	010	0100		
6	011		0111	

Performance of tree protocols

- In case of uniform ID distribution, the tree induced by the query tree is analogous to the tree induced by the BS protocol.
- This is because a set of uniformly distributed tags splits approximately in equal parts at each query, like in the BS protocol.



→ the QT protocol presents the same performance of BS protocol estimated,

How do we measure performance?

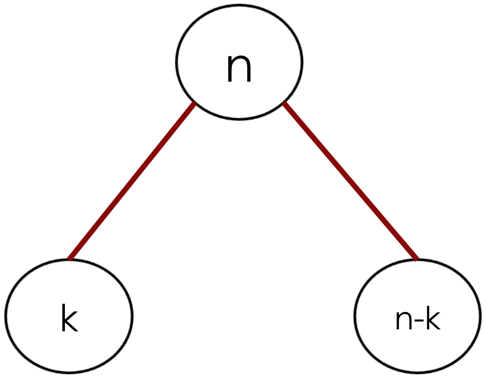
- We want to know how fast a protocol is to collect all tags ID → each tag needs to reply
- If we have n tags, then the protocol will end when all n tags have responded singularly
- System Efficiency $SE = \frac{n}{q}$

where n = single responses, q = total number of queries

- When n=q → optimal protocol
- Unfortunately SE is far below 1!

Performance of Binary Splitting

- To evaluate SE we need to estimate the total number of queries (#Q) that we call $BS_{tot}(n)$
- To evaluate the total number of queries we estimate the **total number of nodes in a BS tree**
- We observe that **at each queries tags split into two sets**
- We **recursively count** the number of nodes in the tree



Performance of Binary Splitting

- We estimate the total number of queries $BS_{tot}(n)$ to identify n tags as
- $BS_{tot}(n) = \begin{cases} 1, \; n \leq 1 \\ 1 + \sum_{k=0}^{n} \binom{n}{k} \left(\frac{1}{2}\right)^{k} \left(1 - \frac{1}{2}\right)^{n-k} (BS_{tot}(k) + BS_{tot}(n-k)), \; n > 1 \end{cases}$
- Evaluating SE function for large values of n we get
- $SE_{BS} = 0.38$
- Only 38% of queries are successful!
- Low efficiency

MAC protocols

Several MAC protocols have been proposed to identify tags in a RFID system

- **Sequential** protocols (aim at singulating tag transmissions)
  - Tree based
    - Binary Splitting
    - Query Tree
    - Variations (Query Tree Improved)
  - Aloha based
    - Framed Slotted Aloha
    - EPC Gen Standard
    - Tree Slotted Aloha
    - Variations (BSTSA)

Aloha principle

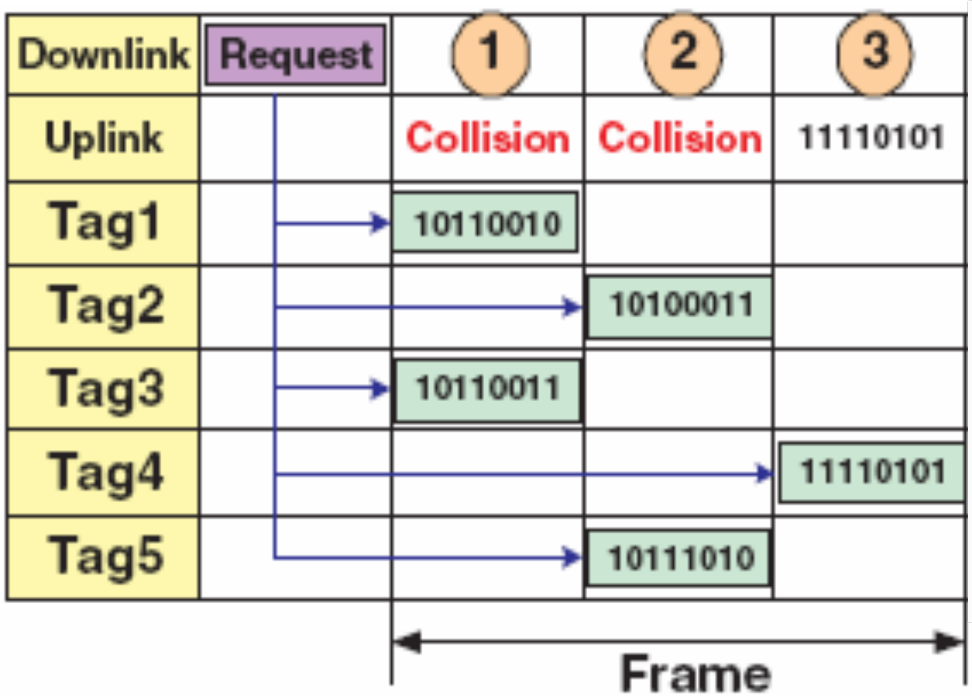
- Time is slotted. Slot duration is equal to the tag's ID transmission time
- Slots are grouped into frames
- Each tag randomly picks a slot to respond

Framed Slotted Aloha (FSA)

- When a reader issues a start of frame, it includes the number of slots in a frame.
- The tags then randomly pick a slot in which to reply.
- Collisions occur, if two or more tags pick the same slot.
- The process repeats itself until all tags are identified.
- Once a tag is identified, it no longer responds to the start of frame

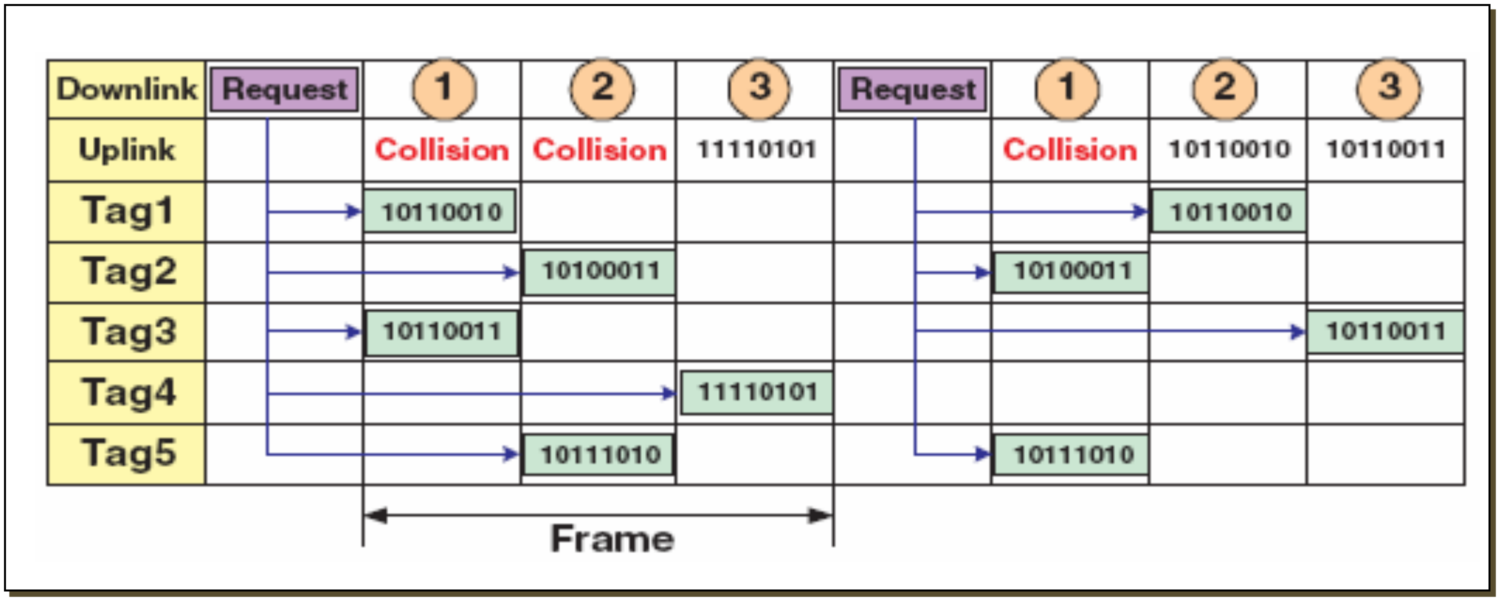
FSA: example

- Slotted Aloha (random selection of slots)



Performance of FSA

- Slotted Aloha (random selection of slots)



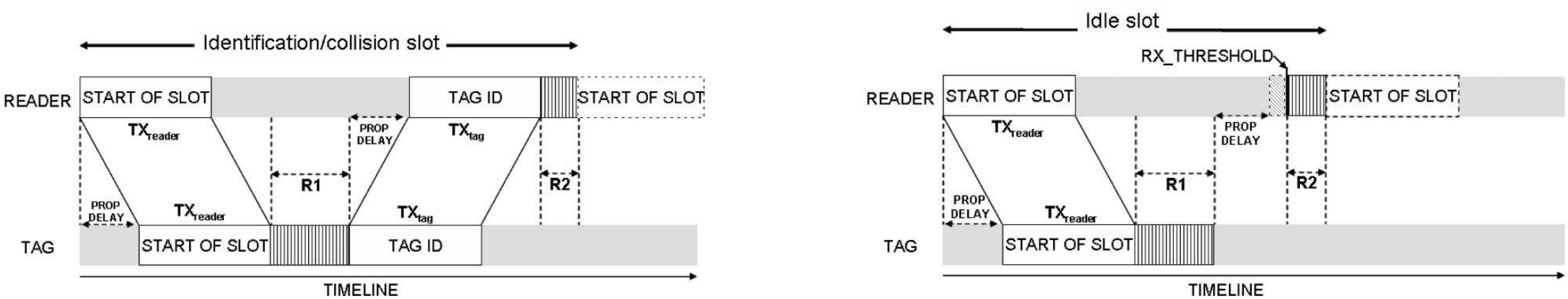
- 6 slots: 3 collisions + 3 identifications
- **System efficiency = # identifications / #slots = 50%**
- In general, best performance is achieved when the number of slots in a frame is equal to the number of tags to be identified
  - 37% of identifications
  - The remaining 63% is wasted in collisions and idle queries

Standard protocol

- The EPC GEN 2 class 1 standard is based on the FSA is the protocol (commercial systems implement the standard protocol)
- EPC adapts frame lenght according to the number of collisions and empty slots
- EPC GEN 2 **specifies the transmission time model** (that allows us to estimate a temporal evaluation of protocol performance)

Transmission time model

- Derived from EPCglobal Specification Class 1 Gen 2

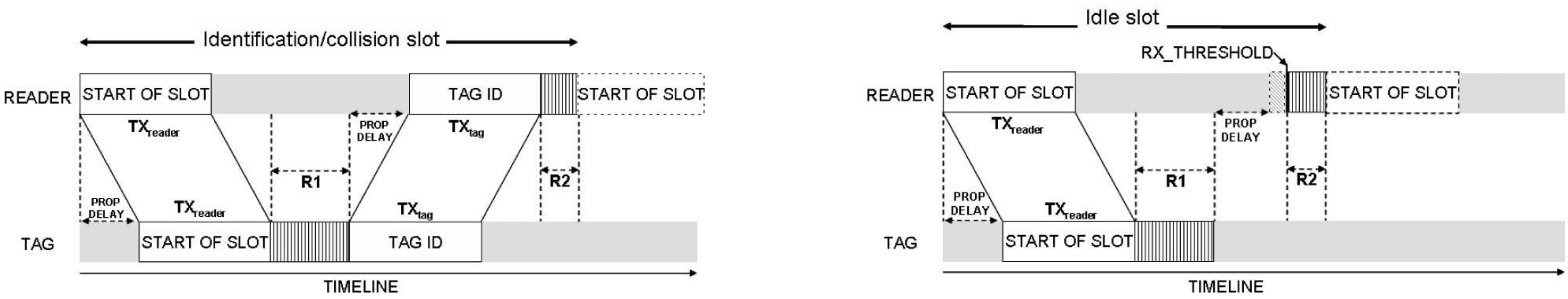


(a) (b)

- R1: tag reaction time
- R2: reader reaction time
- RX_threshold: time at which the reader should receive the first bit of tag transmission

Transmission time model

- Derived from EPCglobal Specification Class 1 Gen 2

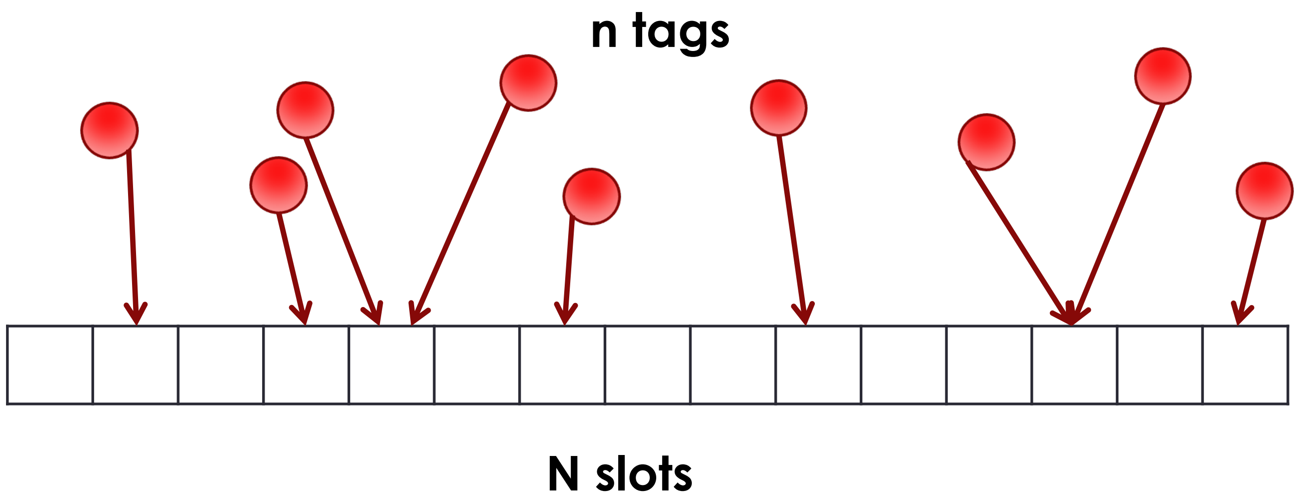


(a) (b)

The **key aspect** of transmission time model stands in observing that idle responses (no response) last less than identification o colliding responses

Analytical model

- Each tag randomly selects a slot



Balls and bins model

Analytical model (cont)

- In Framed Slotted Aloha protocols in which *n* tags randomly select the slot to answer among *N* slots
  - **the probability that *r* tags answer in the same slot** is given by the binomial distribution
  - **The number of slots with exactly r tags is given by**

$$s(r) = N \times \underbrace{\binom{n}{r}\left(\frac{1}{N}\right)^{r}\left(1-\frac{1}{N}\right)^{n-r}}$$

The probability of r out of n tags transmit in one of the N slots

Time system efficiency

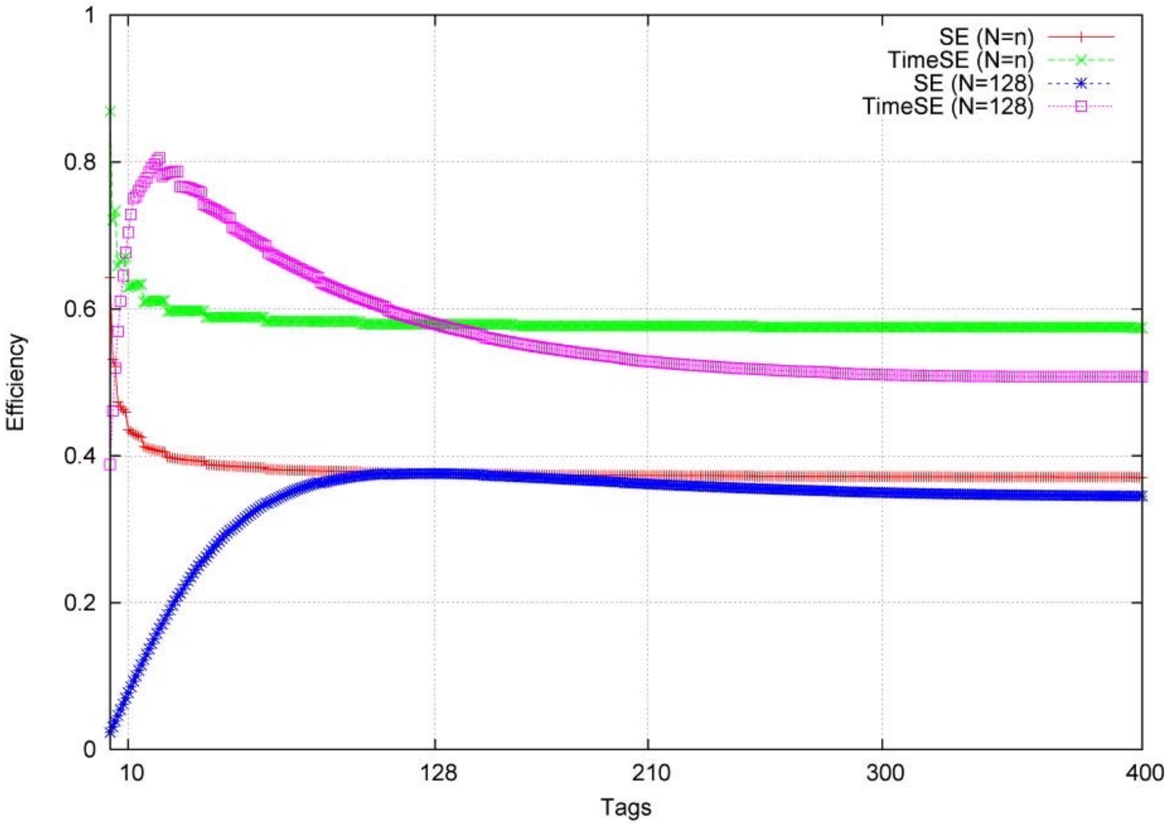
- Let R_{ident}, R_{coll}, and R_{idle} be the number of identification, collision and idle rounds during the tag identification process
- In Framed Slotted Aloha protocols in which n tags randomly select the slot to answer among N slots **the probability that *r* tags answer in the same slot is given by the binomial distribution**
- $R_{idle} = N \times (1 - 1/N)^n$
- $R_{ident} = n \times (1 - 1/N)^{n-1}$
- $R_{coll} = N - R_{idle} - R_{ident}$
- System efficiency = $R_{ident} / (R_{idle} + R_{ident} + R_{coll})$
- In case of rounds of the same duration (weight) is 36%

Time system efficiency

- If idle rounds last a ß fraction of identification and collision round:

$$Time\_SE = \frac{R_{ident}}{\beta R_{idle} + R_{ident} + R_{coll}} = \frac{n\left(1 - \frac{1}{N}\right)^{n-1}}{(\beta - 1)N(1 - \frac{1}{N})^{n} + N}$$

FSA performance: (slots vs time)



When number of tags is known!

System efficiency and time system efficiency for FSA protocol.

FSA performance

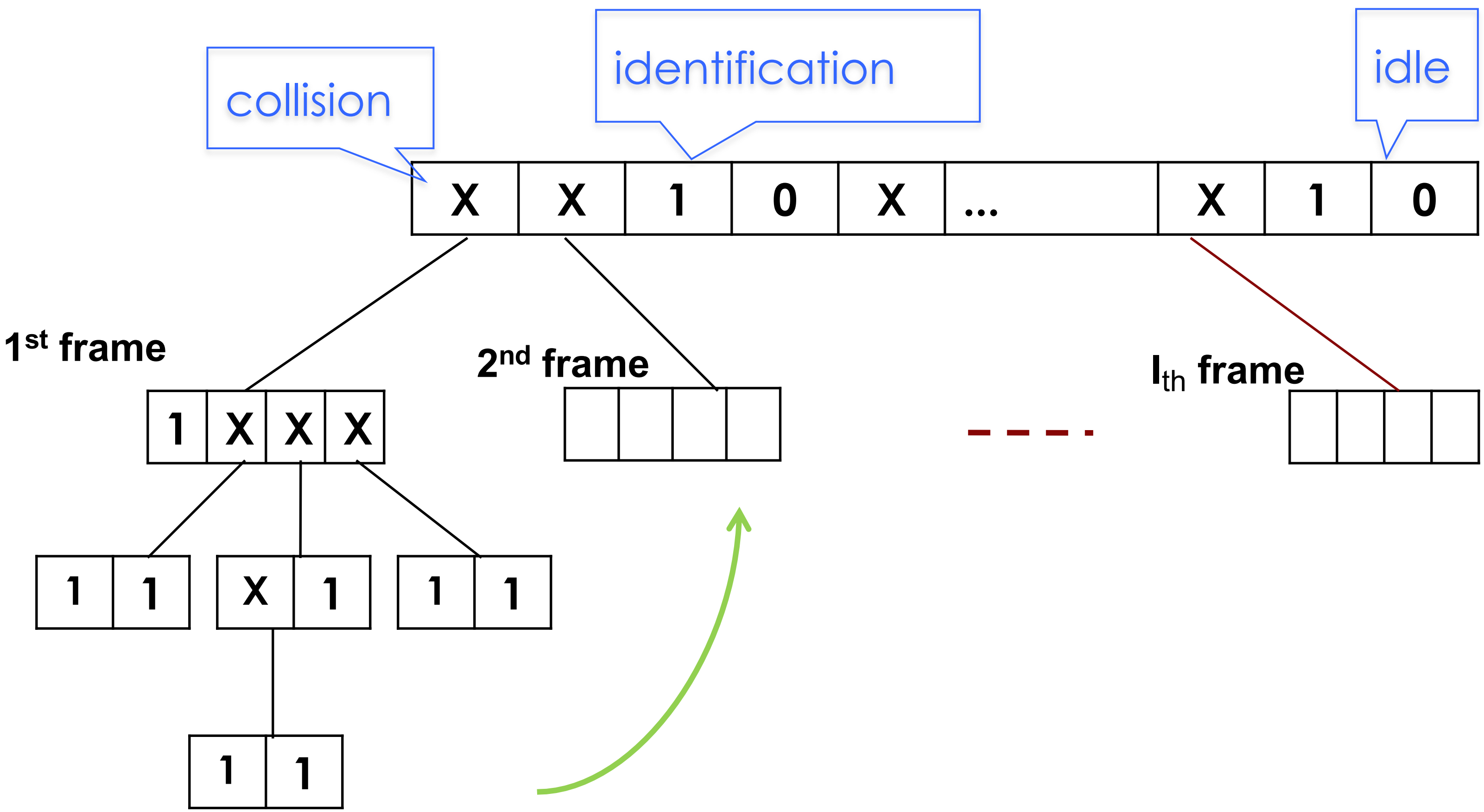
40% of time is waisted in idle and collisions slots!

- **Question**: Can we reduce this time?
- **1rst Answer**: Tree Slotted Aloha

Tree slotted aloha principle

- Slots are executed following a tree
- A new *child* frame is issued for each collision slot: only tags replying to the same slot participate into the new slot

Tree Slotted Aloha (TSA)



TSA performance

- To estimate TSA performance we again **count the number of nodes in the TSA tree**

- $$TSA_{tot}(n) = \begin{cases} 1, \; n = 1 \\ n + n\sum_{k=2}^{n}\binom{n}{k}\left(\frac{1}{n}\right)^{k}\left(1-\frac{1}{n}\right)^{n-k} TSA_{tot}(k), \; n > 1 \end{cases}$$

- For large values of n, $SE_{TSA} = 0.43$ (performance measured in slots)

- While considering different slots duration:

TSA Performance by Optimizing SE (i.e., $N = n$)
and Time_SE (i.e., $N = 4.4 * n - 1$)

Tags	Opt. SE			Opt. Time_SE		
	SE	Time_SE	Latency	SE	Time_SE	Latency
1000	0.38	0.52	6.91	0.28	0.64	5.30
3000	0.37	0.51	21.0	0.20	0.68	14.8
5000	0.37	0.51	35.3	0.20	0.71	23.5

TSA and FSA: main issues

Often the **number of tags** in the system **is not known**

How can we estimate frame size?

Any time TSA issues a new frame it has to estimate the number of tags participating into that frame

And the initial frame? How many tags are in the the environment?

Estimating tag population for intermediate frames

Estimating tag population for intermediate frames

- The number of tags to be identified is not known
- The initial frame size is set to a predefined value (i.e., 128)
- The size of the following frames is estimated

$$\text{tags per collision slot} = \frac{(\text{estimated total num of tags}) - (\text{identified tags})}{\text{collision slots}}$$

- Can we calculate this formula?
- We know number of identified tags and number of collision slots
- But we do not know the total number of tags!

Estimating tag population for intermediate frames

- The total number of tags is estimated according to the outcome of the previous frame (based on Chebyshev's inequality)

$$\text{tags per collision slot} = \frac{(\text{estimated total num of tags}) - (\text{identified tags})}{\text{collision slots}}$$

$$\varepsilon(N, c_0, c_1, c_k) = \min_n \left| \begin{pmatrix} a_0^{N,n} \\ a_1^{N,n} \\ a_k^{N,n} \end{pmatrix} - \begin{pmatrix} c_0 \\ c_1 \\ c_k \end{pmatrix} \right|$$

- N: size of completed frame
- $\langle c_0, c_1, c_k \rangle$ triple of observed values
- $\langle a_0, a_1, a_k \rangle$ triple of estimated values

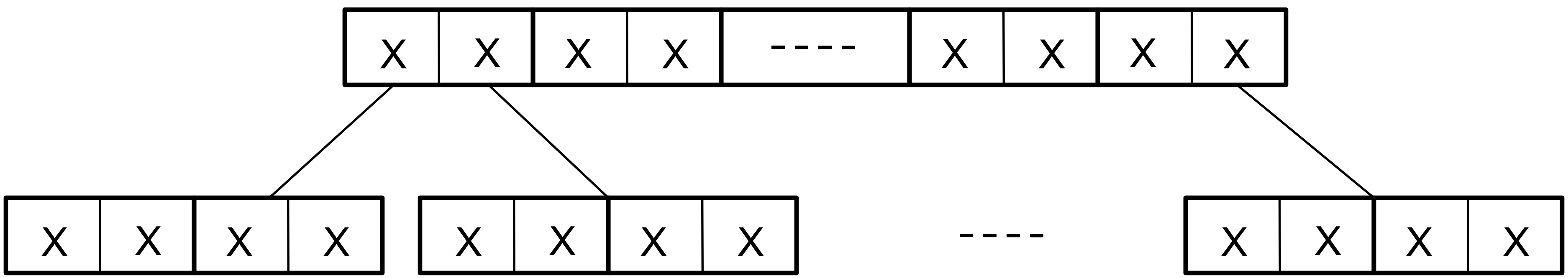
- Given *N* and a possible value of *n*, the expected number of slots with *r* tags is estimated as

$$a_r^{N,n} = N \times \binom{n}{r} \left(\frac{1}{N}\right)^r \left(1 - \frac{1}{N}\right)^{n-r}$$

Inaccuracy of tag estimation for large networks

- The estimator does not capture the possibly high variance of the number of tags
- The minimum distance is computed over n ranging in $[c_1 + 2c_k, 2(c_1 + 2c_k)]$
- The upper bound $2(c_1+2c_k)$ is not adequate for network composed of thousands of nodes
  - Example: 5000 tags, N=128, it is highly likely that c_1=0

    n is estimated $2(c_1+2c_k) = 512$ → definitively too small



Only 4 slots for an expected number of colliding tags around 40!

Unbounded estimator

- Let us search for a better upper bound
- Let us not stop at $2(c_1+2c_k)$
- For N=128 and $<c_0,c_1,c_k> = <0,0,128>$, the table shows the triple of estimated values and their distance from observed value by varying n

Varying n

n	vect. distance	a_0	a_1	a_k
256	64.671	17.187	34.645	76.167
500	16.211	2.536	9.983	115.482
700	4.537	0.528	2.912	124.560
800	2.337	0.241	1.519	126.240
900	1.188	0.110	0.780	127.110
1000	0.598	0.050	0.396	127.554
1500	0.017	0.001	0.012	127.987
2000	0.0005	0.00002	0.0003	127.9997

still not accurate!

Can we find a better solution?

Starting with a proper frame size leads to better estimation also for intermediate frames

How do we estimate the initial tag population?

Estimating initial tag population

We need to estimate the **initial tag population** to properly set the size of the **first** frame

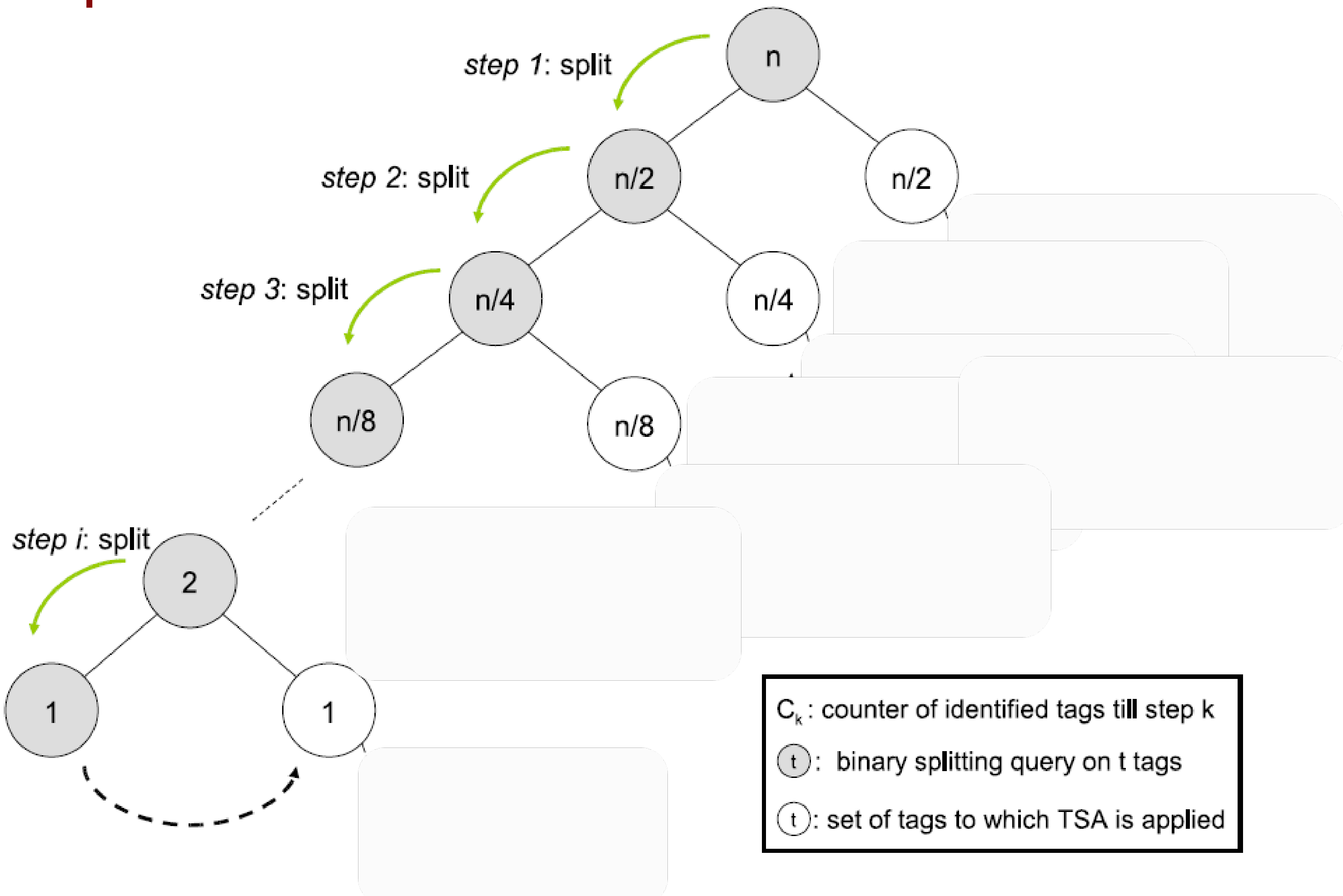
Two solutions

Dy_TSA protocol	**BSTSA protocol**

Binary Splitting Tree Slotted Aloha (BSTSA)

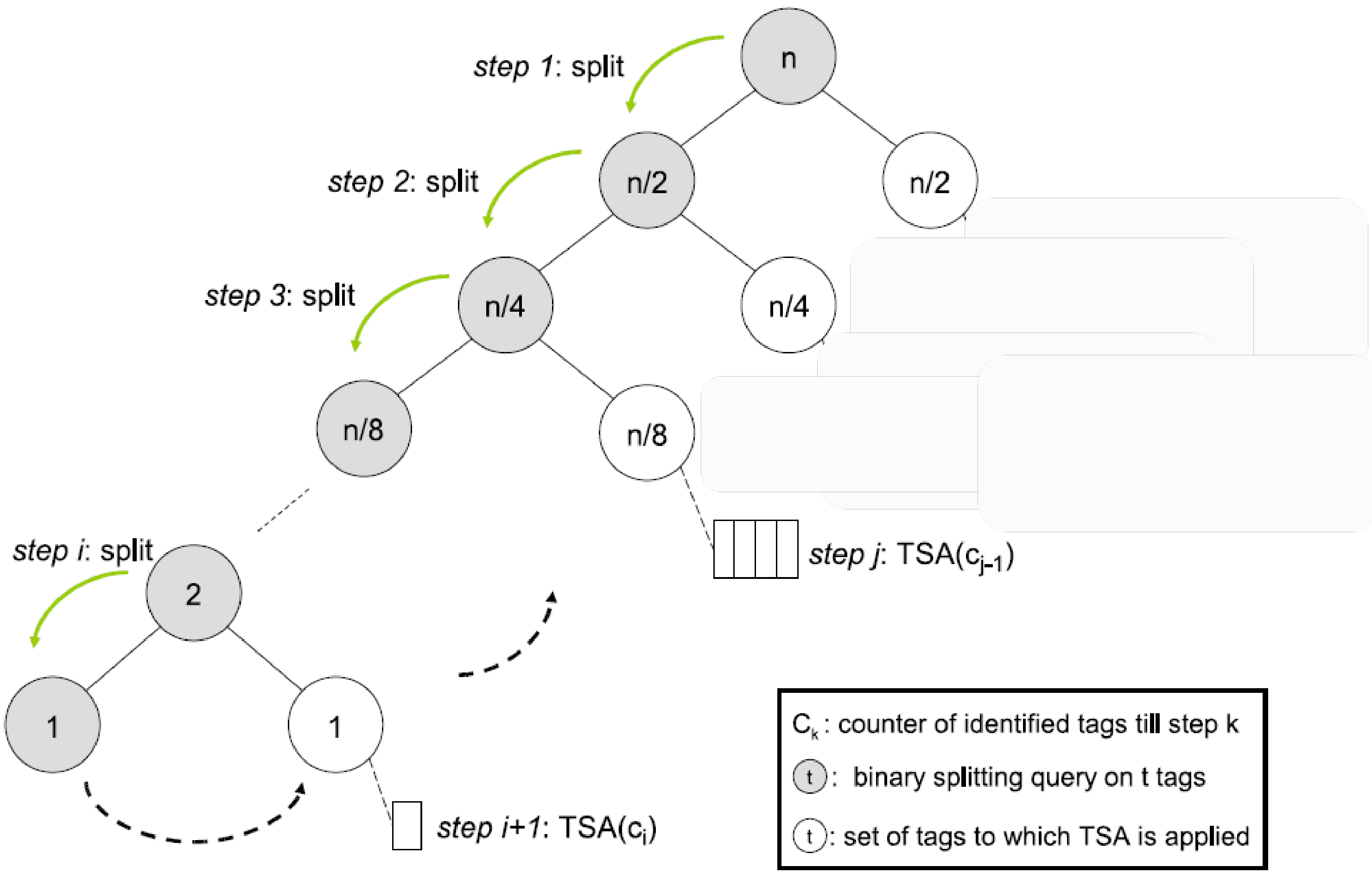
- ***Basic principle*: any large group of elements randomly split into two groups of *almost the same size***
- BS randomly splits tags
- BSTSA: Combination of BS and TSA
  - BS is used to divide tags into groups whose size can be easily estimated
  - TSA is used to identify tags

BSTSA protocol description

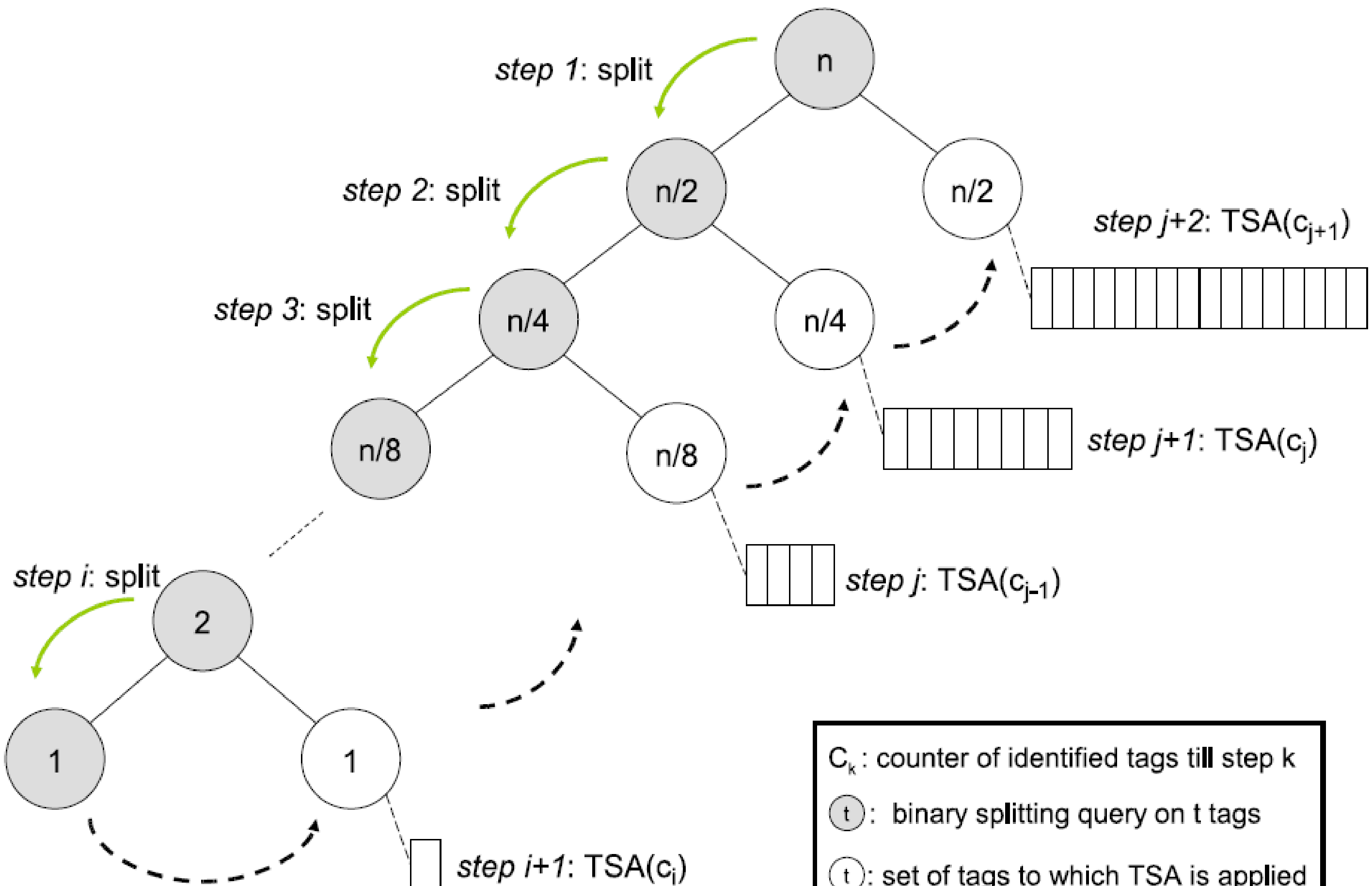


When the splitting process reaches a single-tag group (i.e., the left leaf on the tree), the protocol starts identifying the right siblings on the tree.

BSTSA protocol description



BSTSA protocol description

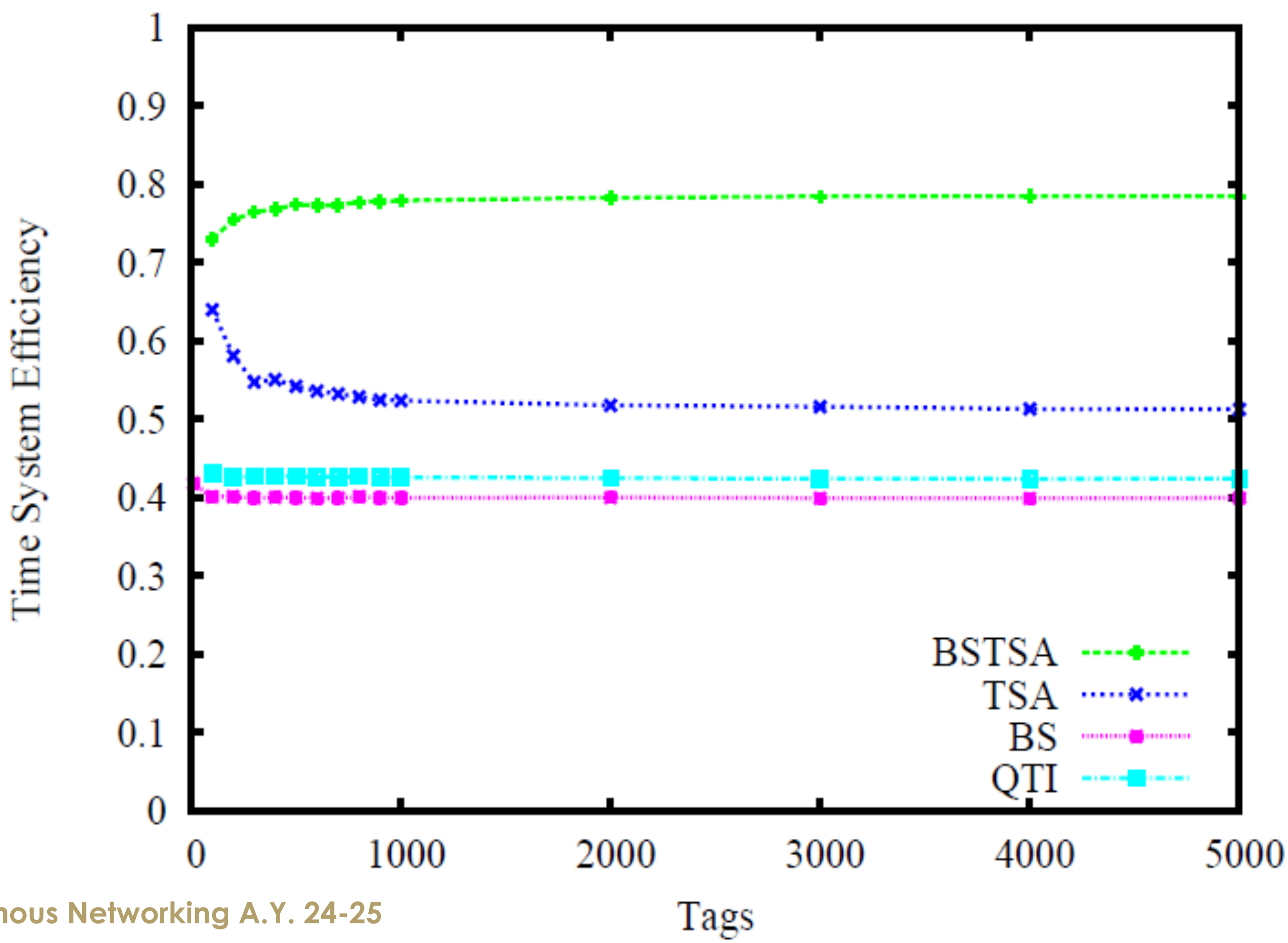


BSTSA performance

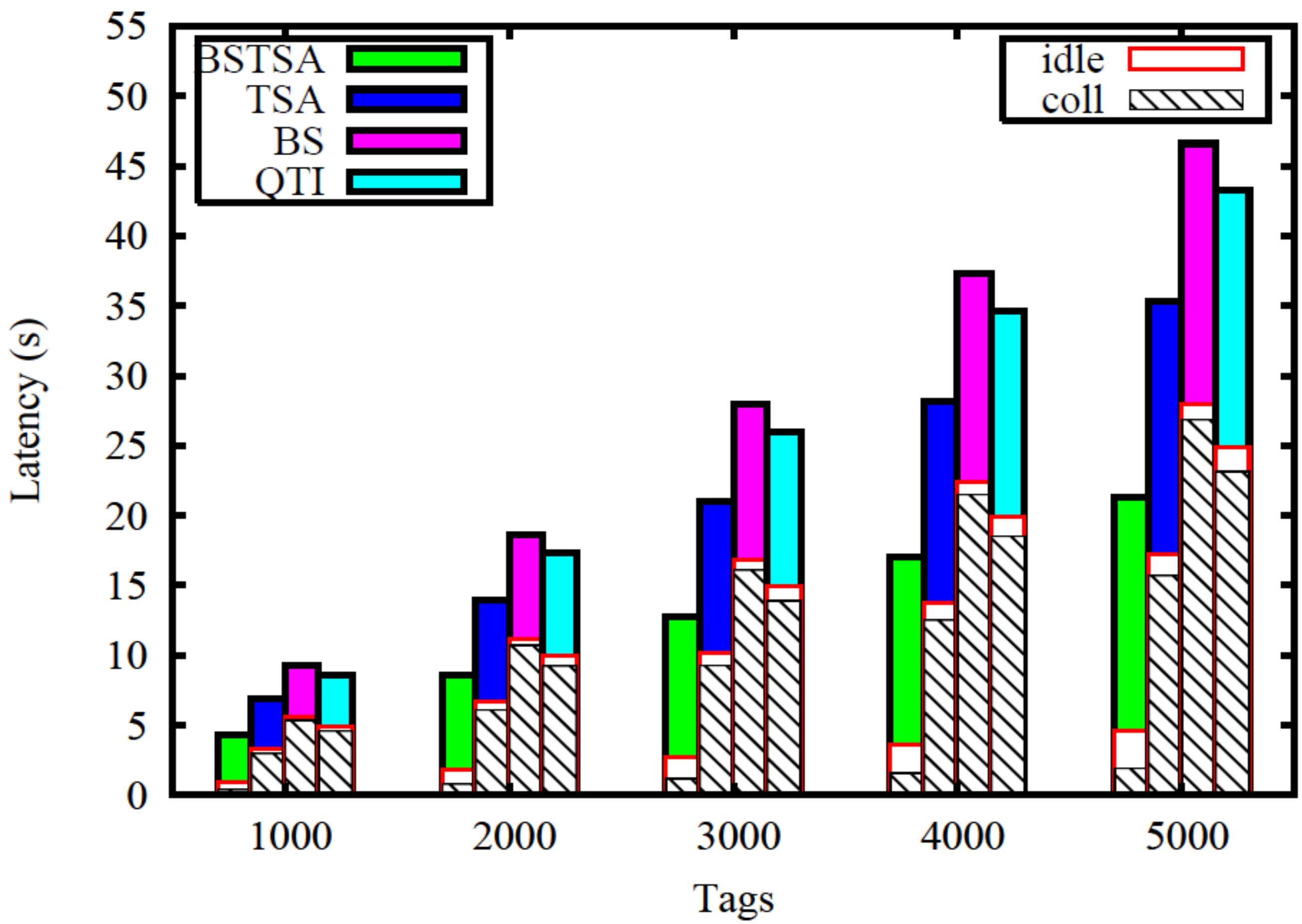
- To evaluate BSTSA performance
  - BS performace up to the last split
  - TSA performance for each group

- Optimal frame tuning is considered (overestimating frame size to allow for more idle slots than collision slots)

Results: Time system efficiency



Results: Latency



Readings

- Paper available on IEEE digital library:
- T.F. La Porta, G. Maselli, C. Petrioli, "**Anti-collision Protocols for Single-Reader RFID Systems: Temporal Analysis and Optimization**", *IEEE Transactions on Mobile Computing*, vol.10, no.2, pp.267,279, Feb. 2011.

Questions?