TUM I 13
Chair of Application and Middleware Systems
Prof. Dr. H.- A. Jacobsen
Author: Pezhman Nasirifard, Kaiwen Zhang

Tutorial
Distributed Systems (IN2259)
Pezhman Nasirifard
WS 2019/20

Issued on: 11.12.2019
Due on: 19/20.12.2019

Sample Solution: Exercises on CAP Theorem, Web Caching & Consistent Hashing

Exercise 1 CAP Theorem

For each of the following scenarios, answer what kind of system is the most adequate for the situation (CA, CP, AP). Justify your answer.

(a) A global monitoring system for air traffic controllers. Each ATC client can request information about any flight. When an anomaly occurs, clients are required to be informed without delay in order to avert disasters. The air traffic controllers are directly connected to a dedicated optical fiber network with a MTBF of 500 years and six-nines availability.

Solution: CA System: Since the network is extremely robust, we can assume no network partitions will occur. Because the system will be handling mission-critical data, it must operate with maximal consistency and availability.

(b) An online shopping website which is specialized in Christmas gifts. It only sells items once at the beginning of each month, at a heavy discount. Items are only guaranteed to be in time for Christmas, no matter when you order.

Solution: AP System: Since it is a normal online site, maintaining P is essential. Between C and A, availability is more important since the website only goes online once a month, so it cannot afford to be unavailable during that time. Because there is a long leeway for fulfilling orders, inconsistent operations can be resolved after the fact. For instance, the system could accept an arbitrary amount of orders, close the sale, then verify which orders can be fulfilled given the amount of stock. Only those orders will be charged, the rest will be canceled.

(c) The local branch of a bank wishes to record all operations performed at its ATMs. Because of security issues, this information cannot leave the physical boundaries of the office (e.g., through a network out of the bank, or with physical media).

Solution: CA System: Since this is a local application, the ATM could rely on a single mainframe computer to log all transactions. This would satisfy the privacy requirements and avoid all network communications. In this case, partitions cannot occur in a centralized system, therefore CA is the appropriate choice.

(d) A promotional offer in a restaurant chain allows customers to use a raffle app on their phone to obtain a prize if their bill exceeds a minimum amount. The grand prize is a new car. The app is controlled by the restaurant company and the promotion operates under a strict global budget which cannot be exceeded. If there are any technical issues with the app, a rain check ticket is given to the customers so that they can come back later to check and claim their prize.

Solution: CP System: Since this is a restaurant chain with a global budget, a distributed system is appropriate. P must therefore be maintained, and the choice is between A and C. Because there is a backup mechanism if the app fails (a rain check), C is the appropriate choice, so that the company doesn't have to rescind the prizes if an inconsistent state occurs. For instance, if multiple cars are handed out when there is only one grand prize. Rescinding a prize would be a PR nightmare. Thus, consistency is important.

Exercise 2 Traditional Hashing vs. Consistent Hashing

A set of N servers (cf. Table 2.2) should be used to cache a collection of M different web sites (cf. Table 2.1) belonging to an application provider. In order to reduce the load on the web server hashing is applied to distribute the web sites among the web caches.
Note: For subtasks (a) and (b) you can neglect the hash values given in the tables.

Name	ID	Hash Value
WebSite_0	0	1A2E
WebSite_1	1	C649
WebSite_2	2	431C
WebSite_3	3	1665
WebSite_4	4	61B3
WebSite_5	5	9271
WebSite_6	6	1CF3
WebSite_7	7	214D
WebSite_8	8	8715
WebSite_9	9	ECA2

Table 2.1: Web sites

Name	ID	Hash Value
WebCache_0	0	7912
WebCache_1	1	CAD4
WebCache_2	2	C23E

Table 2.2: Web caches

(a) In a first approach, the hash function $h(id) = 7 \cdot \text{id} + 4 \text{ mod N}$ is used to associate web sites with caches. Each page is hashed based on its ID and the result of the hash operation is supposed to determine the ID of the corresponding cache. List the set of web sites that is managed by each cache.

Solution:
We have 3 caches so the hash function $h(ID) = 7 \cdot \text{ID} + 4 \; mod \; 3$

Website	ID	$h(ID)$	Webcache
WebSite_0	0	$h(0) = 7 \cdot 0 + 4 \; mod \; 3 = 1$	WebCache_1
WebSite_1	1	$h(1) = 7 \cdot 1 + 4 \; mod \; 3 = 2$	WebCache_2
WebSite_2	2	$h(2) = 7 \cdot 2 + 4 \; mod \; 3 = 0$	WebCache_0
WebSite_3	3	$h(3) = 7 \cdot 3 + 4 \; mod \; 3 = 1$	WebCache_1
WebSite_4	4	$h(4) = 7 \cdot 4 + 4 \; mod \; 3 = 2$	WebCache_2
WebSite_5	5	$h(5) = 7 \cdot 5 + 4 \; mod \; 3 = 0$	WebCache_0
WebSite_6	6	$h(6) = 7 \cdot 6 + 4 \; mod \; 3 = 1$	WebCache_1
WebSite_7	7	$h(7) = 7 \cdot 7 + 4 \; mod \; 3 = 2$	WebCache_2
WebSite_8	8	$h(8) = 7 \cdot 8 + 4 \; mod \; 3 = 0$	WebCache_0
WebSite_9	9	$h(9) = 7 \cdot 9 + 4 \; mod \; 3 = 1$	WebCache_1

Cache	Websites served by cache	#
WebCache_0	WebSite_2, WebSite_5, WebSite_8	3
WebCache_1	WebSite_0, WebSite_3, WebSite_6, WebSite_9	4
WebCache_2	WebSite_1, WebSite_4, WebSite_7	3

(b) How does the situation change when a new server ('WebCache_3', 3, 2F69) is added to the set of caches? Again, list the allocation of web sites to caches. What do you observe? Quantify the degree of reallocation (i.e., the percentage of web sites that need to be transferred).

Solution:
Now, we have 4 caches so the hash function $h(ID) = 7 \cdot \text{ID} + 4 \; mod \; 4$. This results in a total reallocation of web sites to caches.

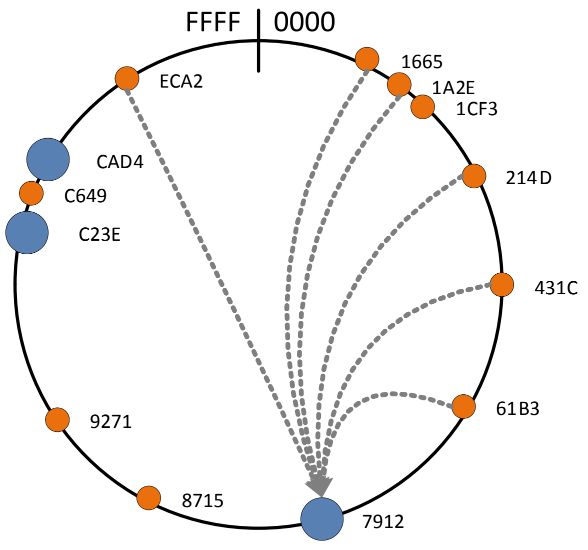
Website	ID	$h(ID) = 7 \cdot \mathrm{ID} + 4 \ mod \ 4$	Webcache
WebSite_0	0	$h(0) = 7 \cdot 0 + 4 \ mod \ 4 = 0$	WebCache_0
WebSite_1	1	$h(1) = 7 \cdot 1 + 4 \ mod \ 4 = 3$	WebCache_3
WebSite_2	2	$h(2) = 7 \cdot 2 + 4 \ mod \ 4 = 2$	WebCache_2
WebSite_3	3	$h(3) = 7 \cdot 3 + 4 \ mod \ 4 = 1$	WebCache_1
WebSite_4	4	$h(4) = 7 \cdot 4 + 4 \ mod \ 4 = 0$	WebCache_0
WebSite_5	5	$h(5) = 7 \cdot 5 + 4 \ mod \ 4 = 3$	WebCache_3
WebSite_6	6	$h(6) = 7 \cdot 6 + 4 \ mod \ 4 = 2$	WebCache_2
WebSite_7	7	$h(7) = 7 \cdot 7 + 4 \ mod \ 4 = 1$	WebCache_1
WebSite_8	8	$h(8) = 7 \cdot 8 + 4 \ mod \ 4 = 0$	WebCache_0
WebSite_9	9	$h(9) = 7 \cdot 9 + 4 \ mod \ 4 = 3$	WebCache_3

Cache	Websites served by cache	#
WebCache_0	WebSite_0, WebSite_4 WebSite_8	3
WebCache_1	WebSite_3, WebSite_7	2
WebCache_2	WebSite_2, WebSite_6	2
WebCache_3	WebSite_1, WebSite_5 WebSite_9	3

Altogether, only 2 web sites out of 10 (i.e., 3 and 8) remain in their cache. This results in 80% ($\frac{10-2}{10}$) of the content that needs to be reallocated.

(c) In a second approach, consistent hashing is used to associate web sites to caches. Similar to MD5, the hash function that is used for this example produces hex numbers comprised of 4 hex digits. Hence, the range of this function is $[0000, FFFF]$. The hash values for caches and web sites are given in Table 2.2 and Table 2.1, respectively. Based on these values associate web sites to the corresponding caches.

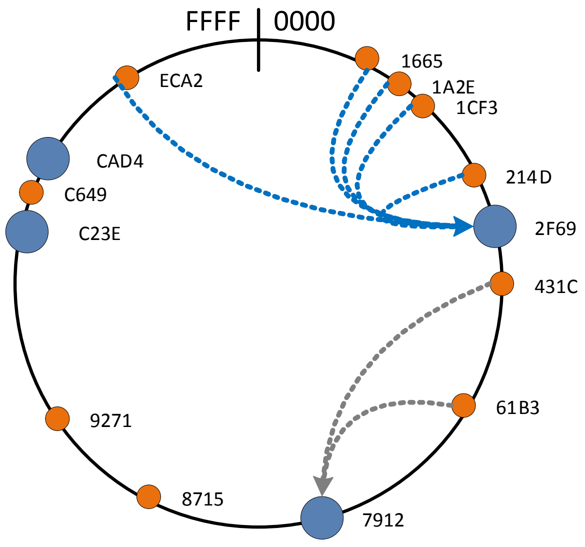
Solution:



Cache	Websites served by cache	#
WebCache_0	WebSite_0, WebSite_2, WebSite_3, WebSite_4, WebSite_6, WebSite_7, WebSite_9	7
WebCache_1	WebSite_1	1
WebCache_2	WebSite_5, WebSite_8	2

(d) How does the situation change when the new server ('WebCache_3', 3, 2F69) is added to the set of caches? Again, list the allocation of web sites to caches. What do you observe? Quantify the degree of reallocation (i.e., the percentage of web sites that need to be transferred).

Solution:



Cache	Websites served by cache	#
WebCache_0	WebSite_2, WebSite_4	2
WebCache_1	WebSite_1	1
WebCache_2	WebSite_5, WebSite_8	2
WebCache_3	WebSite_0, WebSite_3, WebSite_6, WebSite_7, WebSite_9	5

Altogether, 5 out of 10 web sites (i.e., 9, 3, 0, 6 and 7) need to be transferred to the new cache. This results in 50% ($\frac{10-5}{10}$) of the content that needs to be reallocated.

(e) Compare the uniformity of the distribution in subtask (c) and (d) by calculating the standard deviation. What do you conclude from the result? The standard deviation should be calculated with the following formula:

$$S = \sqrt{S^2} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2} \quad ,\text{with} \quad \bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Solution:
For subtask (c):

$$\bar{X}_c = \frac{1}{3} \cdot 10 \approx 3.33$$

$$S_c = \sqrt{\frac{1}{2} \cdot ((7-3.33)^2 + (2-3.33)^2 + (1-3.33)^2)}$$

$$S_c = \sqrt{\frac{1}{2} \cdot ((3.67)^2 + (-1.33)^2 + (-2.33)^2)}$$

$$S_c = \sqrt{\frac{1}{2} \cdot (13.46 + 1.76 + 5.42)}$$

$$S_c \approx 3.21$$

For subtask (d):

$$\bar{X}_d = \frac{1}{4} \cdot 10 = 2.5$$

$$S_d = \sqrt{\frac{1}{3} \cdot ((5-2.5)^2 + (2-2.5)^2 + (2-2.5)^2 + (1-2.5)^2)}$$

$$S_d = \sqrt{\frac{1}{3} \cdot ((2.5)^2 + (-0.5)^2 + (-0.5)^2 + (-1.5)^2)}$$

$$S_d = \sqrt{\frac{1}{3} \cdot (6.25 + 0.25 + 0.25 + 2.25)}$$

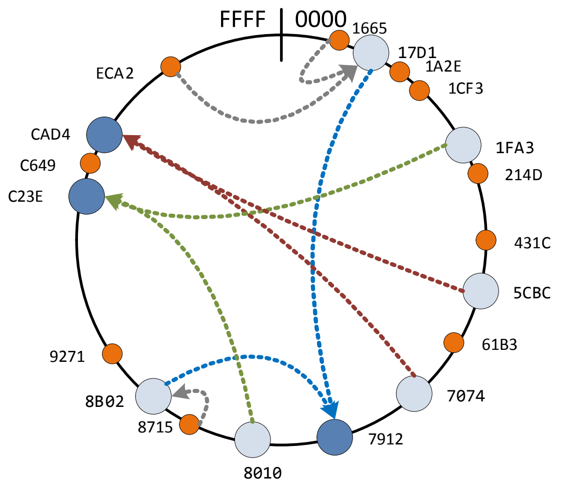
$$S_d \approx 1.73$$

The result shows that with increasing number of caches the standard deviation decreases. This indicates that the distribution of elements/web sites among caches gets more uniform with higher number of available caches.

(f) As we saw, it is possible to have a non-uniform distribution of web sites between caches if there are not enough caches in the system. A possible solution to deal with this problem is to introduce the concept of "virtual nodes", i.e., replicas of web caches, where each real cache corresponds to several virtual caches in the circle. Whenever a cache is added, also a number of virtual nodes is created for the new cache, and when a cache is removed, all its virtual nodes are removed from the circle. However, all objects that are handled by a virtual cache are actually handled by the real cache they are associated with.
For the sake of this example the replication factor should be 2 (i.e., for each cache there are two additional virtual caches). The virtual caches are given in Table 2.3.
Give the allocation of web sites to virtual caches and (real) caches and calculate the standard deviation for the real caches. How does the distribution compare to the above scenarios?

Cache	Virtual Cache	Hash Value
WebCache_0	VirtualCache_0_1	8B02
	VirtualCache_0_2	17D1
WebCache_1	VirtualCache_1_1	5CBC
	VirtualCache_1_2	7074
WebCache_2	VirtualCache_2_1	1FA3
	VirtualCache_2_2	8010

Table 2.3: Web caches and their virtual nodes.



Cache	Websites served by cache	#
WebCache_0		3
VirtualCache_0_1	WebSite_8	
VirtualCache_0_2	WebSite_3, WebSite_9	
WebCache_1	WebSite_1	4
VirtualCache_1_1	WebSite_2, WebSite_7	
VirtualCache_1_2	WebSite_4	
WebCache_2	WebSite_5	3
VirtualCache_2_1	WebSite_0, WebSite_6	
VirtualCache_2_2		

$$\bar{X}_f = \frac{1}{3} \cdot 10 \approx 3.3$$

$$S_f = \sqrt{\frac{1}{2} \cdot ((3-3.3)^2 + (4-3.3)^2 + (3-3.3)^2)}$$

$$S_f = \sqrt{\frac{1}{2} \cdot ((-0.3)^2 + (0.7)^2 + (-0.3)^2}$$

$$S_f = \sqrt{\frac{1}{2} \cdot (0.09 + 0.49 + 0.09)}$$

$$S_f \approx 0.58$$

Due to the (virtually) increased number of caches the probability for an even distribution also increases. Compared to the above scenarios the distribution here is better.

Exercise 3 Reed Solomon Error Correction

A data center makes use of 6 storage servers to store a set of data. The storage manager breaks down the data into shards and stores them across all storage servers. Furthermore, the storage manager utilizes Reed-Solomon encoding for generating and storing parity shards for original data, which are used to recover the corrupted data.

(a) Divide the text "Reed Solomon Err" into four shards and create the parity shards by using the following encoding matrix. (**Hint:** Use the ASCII Table to convert the string to decimal values. `http://www.asciitable.com/`)

$$\begin{bmatrix} 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 01 \\ 27 & 28 & 18 & 20 \\ 28 & 27 & 20 & 18 \end{bmatrix}$$

Solution:
We break the string into four shards and store them in a matrix, where each row present one shard.

$$\begin{bmatrix} R & e & e & d \\ SPACE & S & o & l \\ o & m & o & n \\ SPACE & E & r & r \end{bmatrix}$$

We use the ASCII Table to convert each entry to its corresponding decimal ASCII value.

$$\begin{bmatrix} 82 & 101 & 101 & 100 \\ 32 & 83 & 111 & 108 \\ 111 & 109 & 111 & 110 \\ 32 & 69 & 114 & 114 \end{bmatrix}$$

We multiply the encoding matrix with the converted matrix to calculate the parity matrix. The two last rows of parity matrix are the parity shards which are stored in the storage servers.

$$\begin{bmatrix} 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 01 \\ 27 & 28 & 18 & 20 \\ 28 & 27 & 20 & 18 \end{bmatrix} * \begin{bmatrix} 82 & 101 & 101 & 100 \\ 32 & 83 & 111 & 108 \\ 111 & 109 & 111 & 110 \\ 32 & 69 & 114 & 114 \end{bmatrix} = \begin{bmatrix} 82 & 101 & 101 & 100 \\ 32 & 83 & 111 & 108 \\ 111 & 109 & 111 & 110 \\ 32 & 69 & 114 & 114 \\ \mathbf{5748} & \mathbf{8393} & \mathbf{10113} & \mathbf{9984} \\ \mathbf{5956} & \mathbf{8491} & \mathbf{10097} & \mathbf{9968} \end{bmatrix}.$$

(b) Imagine that the first two shards of the original data are corrupted due to storage server failures. Recover the original data according to the encoding matrix and the stored parity shards.

Solution:
Because the first two shards are corrupted, we need to calculate the inverse matrix of the last four rows of the encoding matrix.

$$\begin{bmatrix} 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 01 \\ 27 & 28 & 18 & 20 \\ 28 & 27 & 20 & 18 \end{bmatrix}^{-1} = \begin{bmatrix} -74/55 & 36/55 & -27/55 & 28/55 \\ 36/55 & -74/55 & 28/55 & -27/55 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

And finally, for recovering the original data, we multiply the calculated inversed matrix with the four last rows of the parity matrix we calculated in the previous part.

$$\begin{bmatrix} -74/55 & 36/55 & -27/55 & 28/55 \\ 36/55 & -74/55 & 28/55 & -27/55 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 111 & 109 & 111 & 110 \\ 32 & 69 & 114 & 114 \\ 5748 & 8393 & 10113 & 9984 \\ 5956 & 8491 & 10097 & 9968 \end{bmatrix} = \begin{bmatrix} 82 & 101 & 101 & 100 \\ 32 & 83 & 111 & 108 \\ 111 & 109 & 111 & 110 \\ 32 & 69 & 114 & 114 \end{bmatrix}.$$

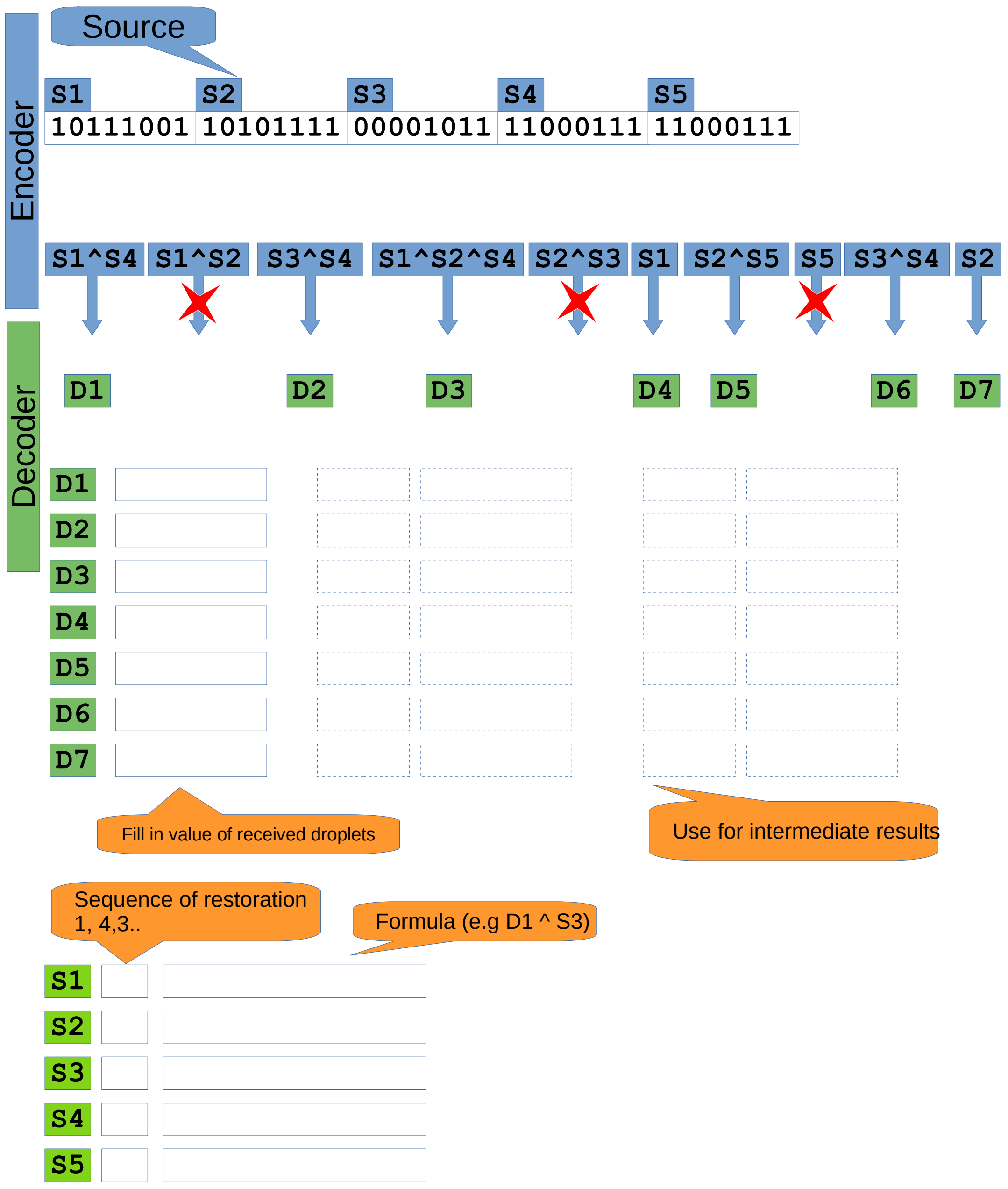
And finally, we use the ASCII Table to convert back the matrix to the original string.

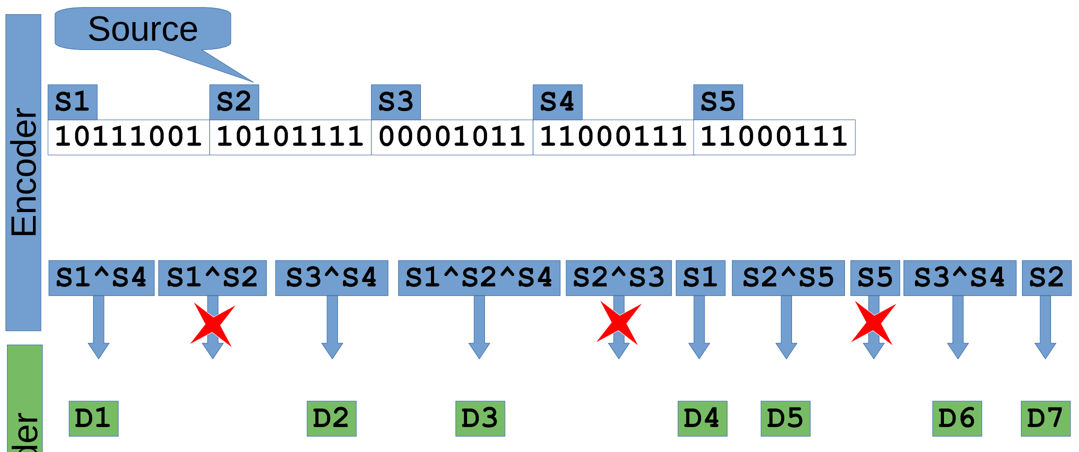
$$\begin{bmatrix} R & e & e & d \\ SPACE & S & o & l \\ o & m & o & n \\ SPACE & E & r & r \end{bmatrix}$$

EXERCISE 4 Luby Transform

Given is a chunk of data which is split into $k = 5$ chunks. The server encodes the data and sends data using UDP to a receiver where it is decoded. The network connection is lossy, some packages drop. The degree of sources and k is in the header of the droplets.

Fill in below the value of the received droplets at the receiver (decoder) side. The boxes can be used for intermediate results. Fill in the sequence of restoration. After which droplet received can the full data be restored?





D1	01111110	D4^D1	11000111		
D2	11001100	D3^S4	00010110		
D3	11010001	D3^S4^S1	10101111		
D4	10111001	S2^D5	11000111		
D5	01101000	D6^S4	00111011		
D6	11001100				
D7	10101111				

Fill in value of received droplets

Use for intermediate results

Sequence of restoration
1, 4,3..

Formula (e.g D1 ^ S3)

S1	1	D4
S2	3	D3^S1^S4
S3	5	S4^D6
S4	2	D4^D1
S5	4	D5^S2

After which droplet received can the full data be restored? D5