LULEÅ TEKNISKA UNIVERSITET

HEMTENTAMEN I **D7001D Network programming and distributed applications** (LP1 2017)

Exam date

Mini-project: A special kind of data storage

LYCKA TILL!

For the home exam you are requested to apply all the skills and technologies you mastered during the course and realize a mini project as per the following description.

You will design a *toy*[1] distributed networking system implementing a special kind of storage. This storage will back up and restore chunks of ***X*** bits of data, where ***X*** is a configurable parameter, which can take **any** values. Your implementation will be tested on X equal 1000 bit, 10000 bits, 100000 bits. The size of data chunks is configured during system initialization and remains unchanged during the system's lifetime. You could think about it as about the fixed length of the packet in your system.

Architecturally the storage system is a hybrid peer-to-peer network with peer nodes that will preform the actual task of storage and reconstruction of data and the infrastructure nodes implementing helper functions. Your software architecture should be able to support a potentially large number of peers ***P*** (***P*** is in the order of 10^6 – one million – nodes being active simultaneously).

The data to be stored (further referred as to Data) are randomly generated sequences of ***X*** bits chunks. The system should be tested on storing 10, 000 chunks of Data. The Data can enter the system from any peer node; therefore a peer should realize a Graphical User Interface for entering the data. Important: the GUI should also implement a button for generating Data chunks. The entered Data will be stored in several nodes (possibly even in the entering node itself) based on the criteria specified below. Each peer will have a capacity of storing up to ***N*** chunks. You should experimentally discover the value of ***N*** providing the best performance.

The criteria for storing Data:
The entire peer-to-peer system will maintain a pool of addresses for storage locations. The number of storage locations across the system is large (more than 10^6). This global number of storage locations (denoted as ***S=PN***) is a configurable parameter via ***P*** and ***N***.

Each storage location is ***X*** bins in size and is addressed by an ***X*** bits address, where ***X*** is the size of the data chunk. The addresses are ***X***-bit binary sequences, which are RANDOMLY generated during the peer's initialization time and remain stable during the peer's lifetime, i.e. each peer has an ***N***x***X*** matrix for addressing purposes. Note that at the initialization time all storage locations are empty.

The Data will be stored based on the principles of proximity/similarity between the Data chunk and the address. The proximity metric is Hamming distance (the number of bit positions in which two sequences disagree). The threshold ***T*** (in number of different bits) is a configurable parameter of the system.

The incoming Data is stored only in these storage locations for which Hamming distance between Data and the location's address is less or equal ***T.*** Thus, Data is stored in many (in the order of tens of thousand) locations on different peers. This implies that the storage request will be propagated through the network and potentially stored in many peers. For

[1] The exam grade of the FIRST two students who will present a proof that this is not a toy storage system but an existing approach will be increased by 1.

this to work you have to implement a broadcast strategy, where the request is propagated to all active peers.

The procedure for storing Data: The data will be stored in a somewhat unconventional way. It will be POSITIONWISE added to the content of all identified storage locations. It is probably wise to use byte array of size **X** for implementing each storage position. Thus, as you can see each storage location will potentially contain a superposition of many stored Data chunks. A hint: While the Data is binary in nature, for the ease of computations at this step encode '0' as "-1" and '1' as "+1", i.e. convert your data from binary format to bipolar format. You also should implement a configurable threshold (***C***), constant for each position of the storage. The value of each position should be in the range [-***C***, +***C***].

The search procedure: The architecture you will design is best suited for the best match search. This means that the query to the system is a Data chunk of size ***X*** bits, which is similar to one of the stored Data chunks. The proximity is again measured by Hamming distance.

The query can be generated at any peer and the network should cooperatively find the best matching stored Data. To do this the querying peer will perform several iterations monitoring Hamming distance between the response of the iteration i-1 and the current iteration. The distance will either converge to 0 or diverge to ***X*/2.** This is your search TERMINATION criteria.

On each iteration the query is broadcasted to all peers. Each peer selects the content of all matching storage locations if any (i.e., Hamming distance between the query and an address is less or equal ***T***), performs a position-wise summation (no thresholding at this step) and sends the result to the requesting peer. Since more than one peer can contain the matching record, several responses can arrive to the requesting peer. These several responses will be treated according to the following procedure:

1. all individual responses will be summed position-wise
2. the resultant sum will be binarized: all positive positions including 0 will become 1 and all negative positions will become 0

This binarized data chunk will be the RESPONSE for a single iteration.

The RESULT of the query is either the binarized Data chunk from the latest iteration if the search procedure has converged or an error code if the procedure has diverged (see the TERMINATION criteria above).

That's it. Now here are the criteria for grading the exam:

1. To get the HIGHEST grade – 5: well documented code, fully implemented functionality, passed tests (to be provided later) + clean *.PPT presentation describing the design of the system and performance evaluation
2. To get 4: working implementation of the core functionality, but with hard coded parameters, satisfactory performed tests and the PPT presentation of the design
3. To get 3: partially working solution implementing the key networking functionality, PPT presentation describing the design and encountered difficulties and the ways to overcome them.

GOOG LUCK!