A Swarm-based Semantic Storage Service

Freie Universität Berlin

Hannes Mühleisen, Kia Teymourian and Robert Tolksdorf

muehleis@inf.fu-berlin.de kia@inf.fu-berlin.de tolk@ag-nbi.de

Networked Information Systems (NBI)

www.ag-nbi.de

Motivation

The amount of data handled by semantic applications is expected to increase over a level manageable by available storage systems. Distributed semantic storage solutions are a powerful concept to increase storage capacity. We are in the process of developing a **Self-Organized Semantic Storage Service (S4)** using swarm intelligence to overcome present limitations in storage capacity and network dynamics.

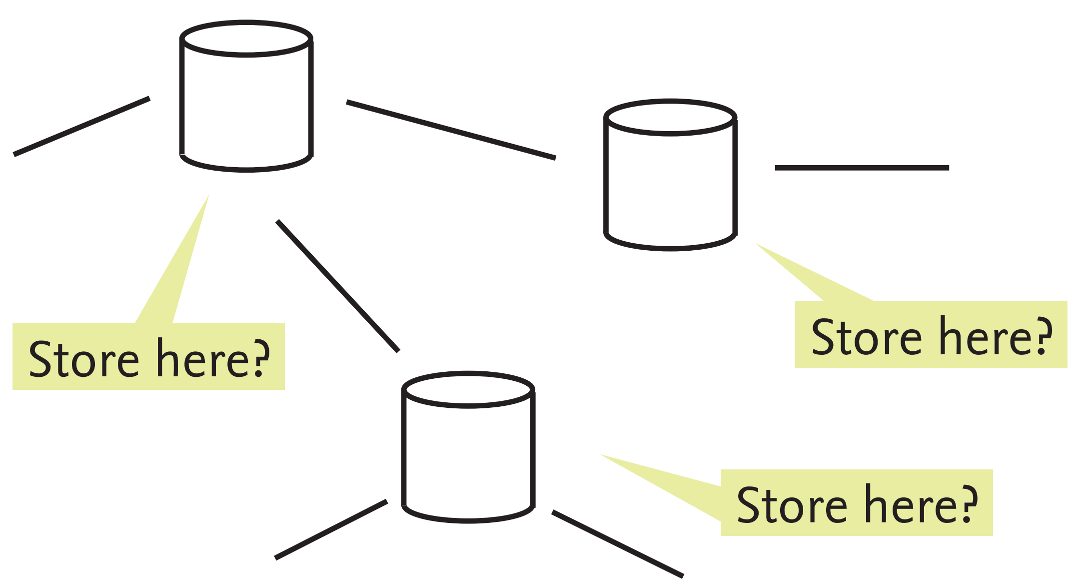
Swarm Intelligence

Decentralized behaviour of large numbers of individuals. Example: **Ant Colonies** perform "Foraging" for efficient food retrieval and "Brood Sorting" to organize their offspring.

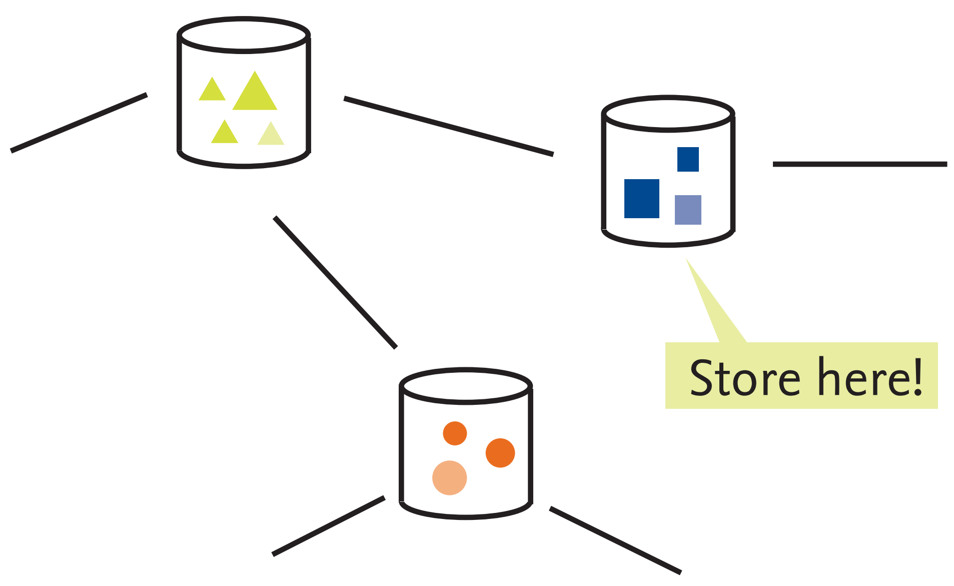
Swarm algorithms make use of many virtual individuals with limited memory, limited view, and limited lifespan. We have modeled our storage operations to "be" cooperating swarm individuals.

Data Storage

Task: write (■) - Where should "■" be stored?



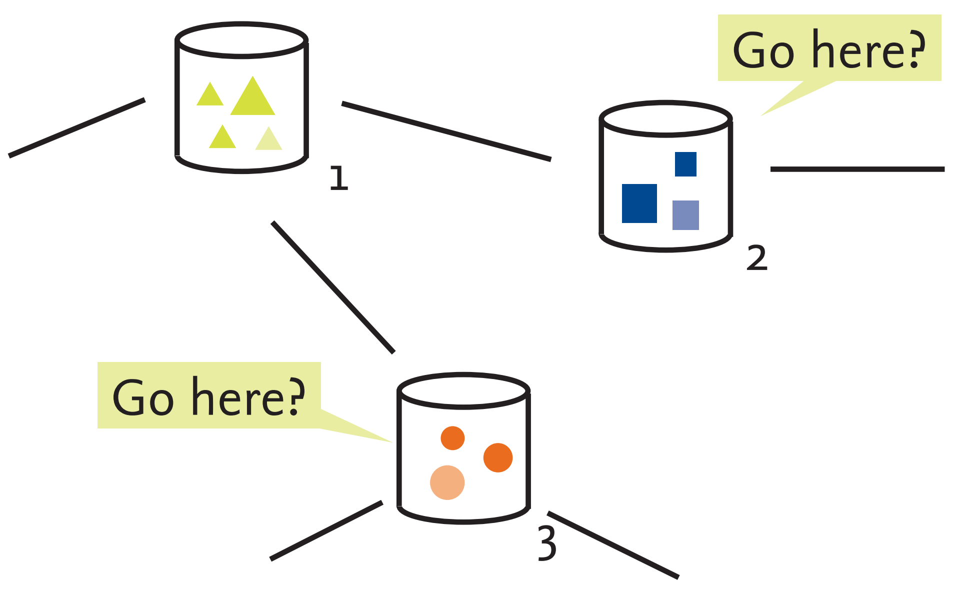
Swarm Solution: Store **similar data** items on the **same node**



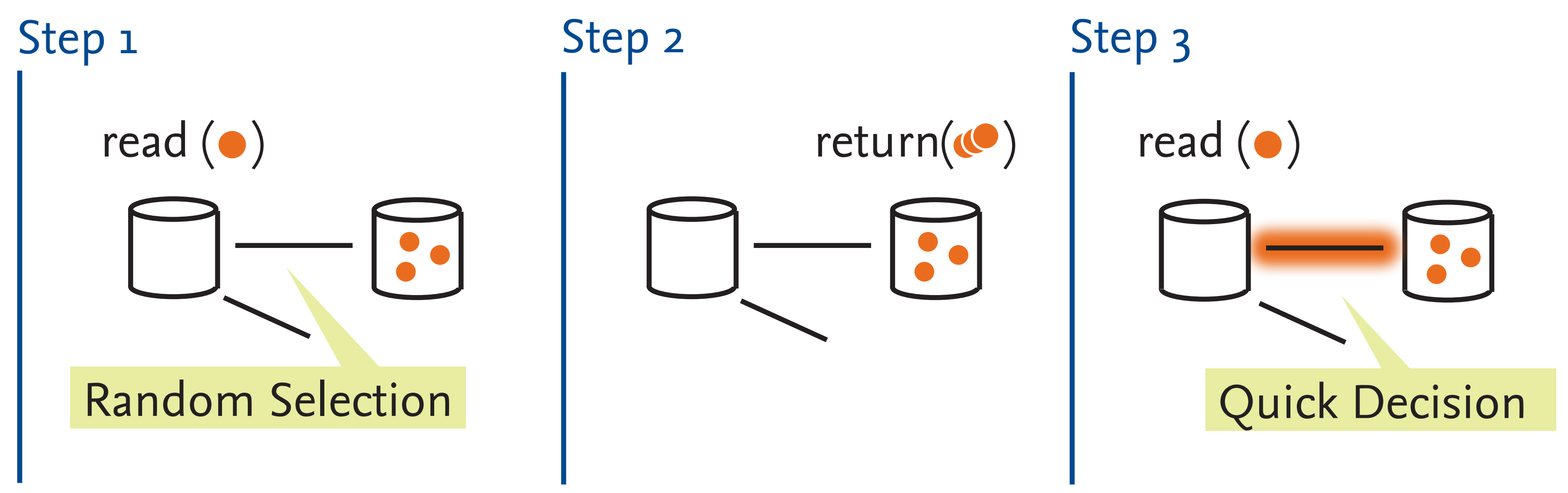
Swarm individuals move data items until they find a node where the carried item **fits in.**

Data Lookup and Retrieval

Task: read (●) on node 1 - How to find "●"?



Swarm Solution: Use **virtual pheromones** to mark paths



Swarm individuals **follow** pheromone trails left by previous operations.

RDF Storage and Retrieval

RDF Triples are stored in the S4 system by using each triple element as a storage key for virtual swarm individuals:

```
store(S,P,O) ->
    write(S,(P,O)); write(P,(S,O)); write(O,(S,P));
```

SPARQL queries are evaluated by determining the basic graph patterns within the query and dispatching an individual for each defined component:

```
SELECT ?r WHERE {?r o:p1 'aVal'} ->
    read(o:p1); read('aVal');
```

Similarity Metrics

Task: Define "similarity" for RDF resources:

1) **Syntactic** similarity metric

+ Easy to compute
- Loss of semantic information

```
sim(<p:foo/1>,<p:foo/2>) >
sim(<p:foo/1>,<p:bar/5>)
```

2) **Semantic** similarity metric

+ Chance to exploit RDF structures
- Ontology has to be known

```
sim('cat','dog') > sim('cat','car')
```

STI · BERLIN

SPONSORED BY THE

Federal Ministry of Education and Research

DigiPolis - 03WKP07B