DASNG

Deep Associative Semantic Neural Graphs
for Knowledge Representation and Fast Data Exploration

AGH University of Science and Technology

Adrian Horzyk

horzyk@agh.edu.pl

Google: Horzyk

Brain-Like Associative Processes

can be used to organize and associate data in deep neural structures…

Objectives and Contribution

- Implementation of associative mechanisms inspired by brains.
- Construction of deep associative semantic neural graphs DASNG for associative representation of the data stored in relational databases.
- Introduction of a new associative spiking model of neurons that can quickly point out related data and entities and be used for inference.
- Innovation in data storage, organization, access, and management that combines, integrates, aggregates & associates various data collections.
- Implementation of a new mechanism of data access and data processing.
- Efficient representation of wider range of data relations directly in the structure, especially horizontal and vertical relations between entities.
- Replacement of time-consuming procedures by the associative structure which significantly reduces the computational complexity of various operations on data and entities, especially of the search operations.

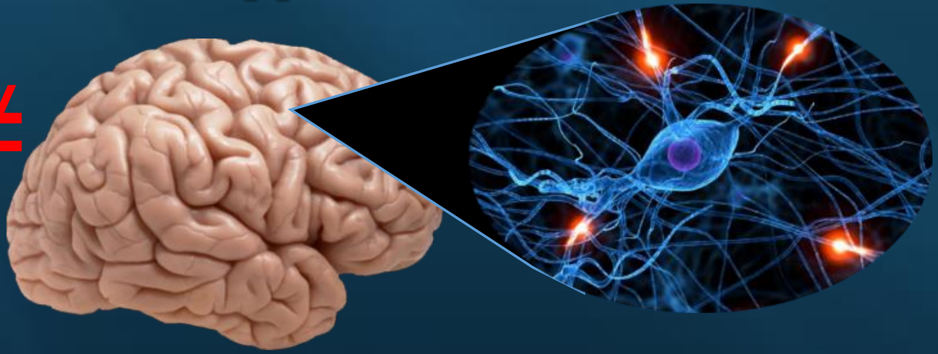
Limitations of contemporary computers

Contemporary computers:

- are limited by **the limitations of the Turing machine** computational model,
- use **array RAM** hindering the implementations of neural graphs,
- **separate the data from the program** and **the memory from the CPU or GPU,**
- execute instructions **sequentially,**
- use **synchronous parallelism** in the GPUs which does not go hand in hand with the way the neurons work in brains.

Such an environment is not beneficial for simulating asynchronous neurons in brain-like graph structures which use a time approach!

≠



Brains and Neurons

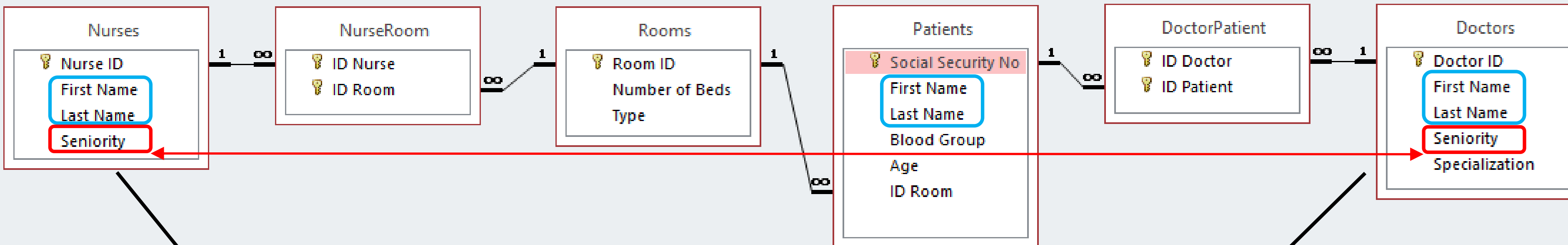
- ✓ execute stimulations **parallel** and often **asynchronously,**
- ✓ **automatically, fast and context-sensitively associate** data and entities,

- ✓ use a complex **graph** memory structure and parallel procedures,
- ✓ **integrate the memory with the program** which use previous **knowledge,**
- ✓ **use time approach** for **temporal** and **contextual** computations,
- ✓ are not limited by **the Turing machine** computational model.

Selected Drawbacks of Relational Data Model

AGH

✓ Representation of horizontal relations between entities (objects)



Nurses

Nurse ID	First Name	Last Name	Seniority
N1	Amy	Moon	12
N2	Rose	Jolie	18
N3	Kate	Ford	24
N4	Lisa	Brown	9
N5	Sara	Pitt	4
N6	Kate	Lopez	12

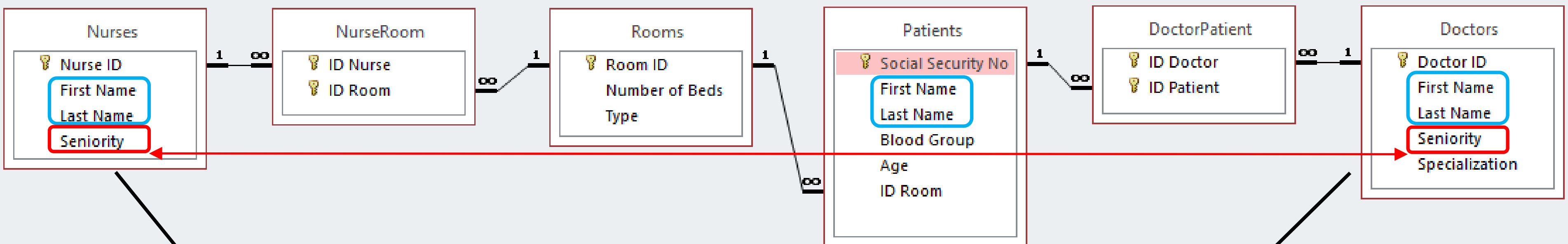
Doctors

Doctor ID	First Name	Last Name	Seniority	Specialization
D1	Tom	Hanks	18	orthopedics
D2	Jack	Brown	15	surgery
D3	Lisa	Ford	23	pediatrician
D4	Tom	Trump	35	pediatrician
D5	Kate	Smith	7	surgery
D6	Amy	Hanks	12	surgery

1. The lack of representation of vertical relations between objects in each table.
2. The necessity to find out vertical relations between objects as order, similarity...
3. The more entities are stored in the table the bigger problem we have (BIG DATA).
4. Non-efficient representation of the duplicated data in the same or various tables.
5. Non-associated parameters and data in various tables describing the same categories.

Consequences of the Drawbacks Relational Data Model

✓ Representation of horizontal relations between entities (objects)



Nurses

Nurse ID	First Name	Last Name	Seniority
N1	Amy	Moon	12
N2	Rose	Jolie	18
N3	Kate	Ford	24
N4	Lisa	Brown	9
N5	Sara	Pitt	4
N6	Kate	Lopez	12

Doctors

Doctor ID	First Name	Last Name	Seniority	Specialization
D1	Tom	Hanks	18	orthopedics
D2	Jack	Brown	15	surgery
D3	Lisa	Ford	23	pediatrician
D4	Tom	Trump	35	pediatrician
D5	Kate	Smith	7	surgery
D6	Amy	Hanks	12	surgery

1. We have to use various **search routines** (inside SELECT) **to retrieve information**.
2. Each search routine **costs time and power** because they use many **nested loops**.
3. The results of the work are often **unsaved or non-suitable** for further operations.
4. We must store many entities of **duplicates** which are **not aggregated** in this model.
5. We must **use indices** because parameters are **not ordered or related** due to their values.

Associative Transformation of Relational Database

Patients

Social Security No	First Name	Last Name	Blood Group	Age	ID Room
17102595834	Jack	Hanks	0	0	R6
17102785603	Nina	Rock	AB	0	R6
73040593020	Tom	Kite	A	44	R4
84021205010	Tom	Ford	AB	33	R1
90503029943	Emy	Cruise	A	27	R2
94051382054	Lisa	White	B	23	R3
94080639502	Paula	Smith	B	23	R2

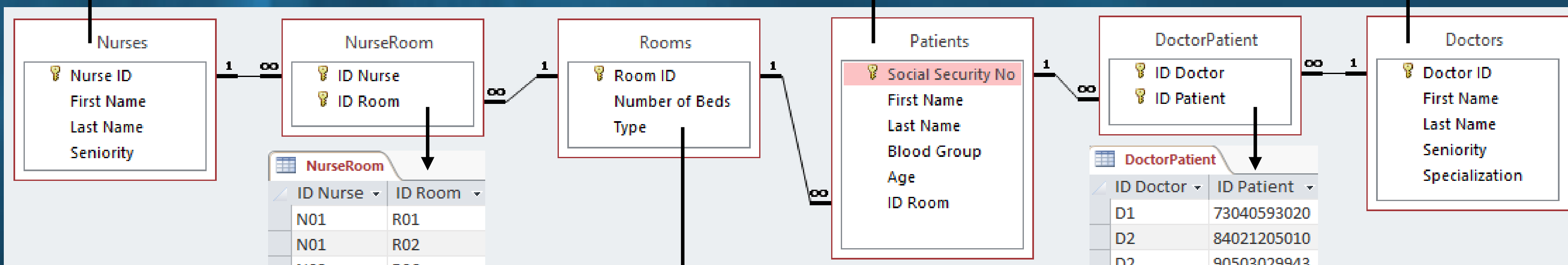
Nurses

Nurse ID	First Name	Last Name	Seniority
N1	Amy	Moon	12
N2	Rose	Jolie	18
N3	Kate	Ford	24
N4	Lisa	Brown	9
N5	Sara	Pitt	4
N6	Kate	Lopez	12

Small hospital database

Doctors

Doctor ID	First Name	Last Name	Seniority	Specialization
D1	Tom	Hanks	18	orthopedics
D2	Jack	Brown	15	surgery
D3	Lisa	Ford	23	pediatrician
D4	Tom	Trump	35	pediatrician
D5	Kate	Smith	7	surgery
D6	Amy	Hanks	12	surgery



NurseRoom

ID Nurse	ID Room
N01	R01
N01	R02
N02	R06
N03	R03
N03	R04
N03	R05
N04	R01
N04	R02
N05	R03
N05	R04
N05	R05
N06	R06

Rooms

Room ID	Number of Beds	Type
R1	2	intensive care
R2	2	intensive care
R3	3	therapeutic
R4	3	therapeutic
R5	3	therapeutic
R6	8	newborns

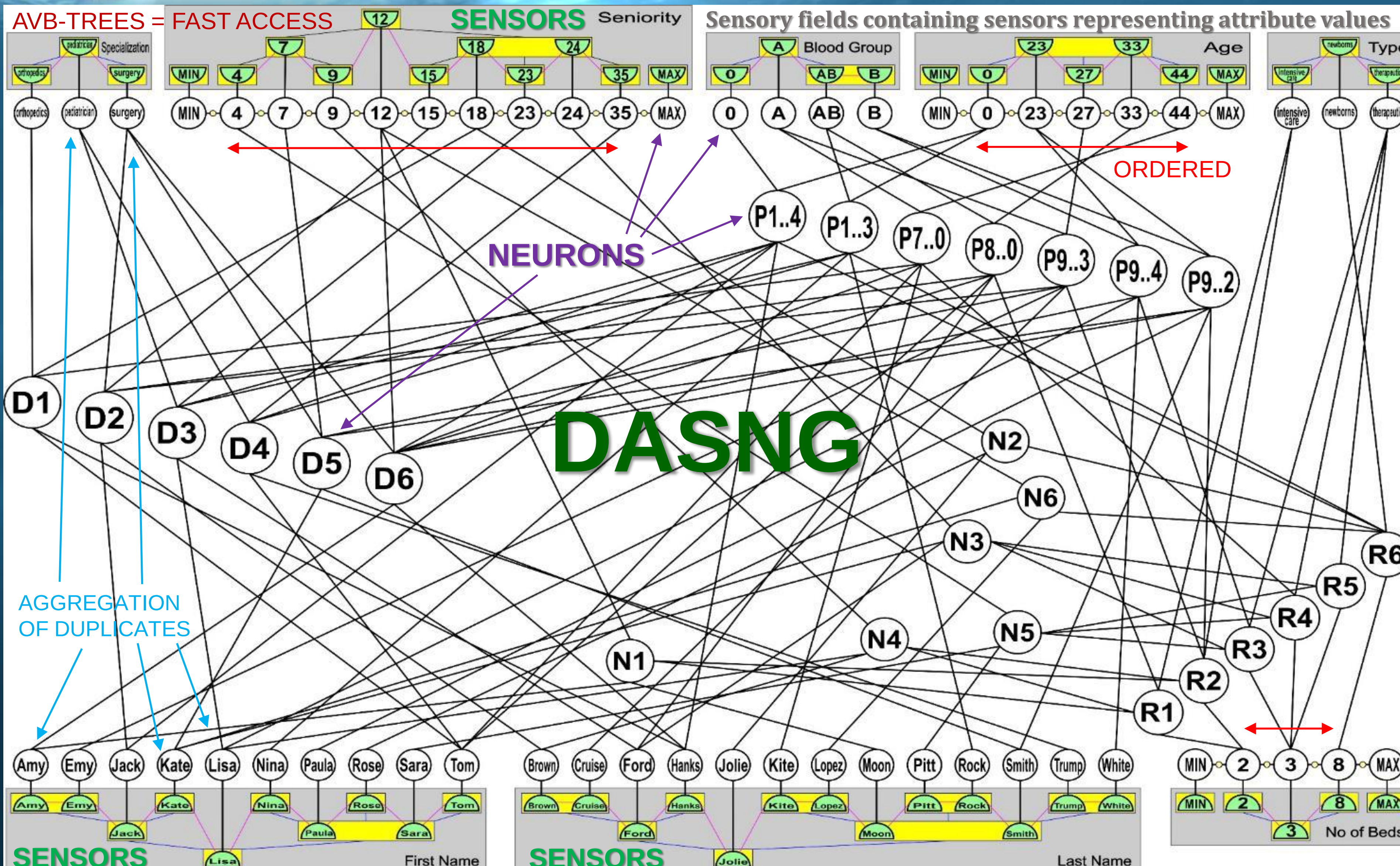
DoctorPatient

ID Doctor	ID Patient
D1	73040593020
D2	84021205010
D2	90503029943
D3	17102595834
D3	17102785603
D4	17102595834
D4	17102785603
D5	90503029943
D5	94080639502
D6	17102595834
D6	73040593020
D6	94051382054
D6	84021205010
D6	94080639502

Replacement of Search Operations by the Associative Graph Structure

- In order to accelerate search routines, we should associate all related data and entities (objects) represented in the database, namely:
  - Sort all orderable attribute values,
  - Directly connect related objects by link-tables,
  - Aggregate all the same values (duplicates) of the same categories.
- In consequence, related objects will be quickly available and will not require to be searched, indexed, or compared in many nested loops.
- All duplicated values of the same category occurring in the same or different tables will be aggregated and sorter.
- In result, we achieve an associative graph structure representing all horizontal and additionally vertical relations between data and objects.
- The graph nodes contain the numbers of aggregated duplicates.
- The graph connections contain the information about the strength of relations of the connected objects or values.

Associative Graph Structure Can Replace Many Search Operations



DASNG – Deep Associative Semantic Neural Graphs

- **D – deep** – means the ability to represent various data relations in the specific deep neural network structure.
- **A – associative** – stands for the way human brain works, it allows for the fast availability of various data accordingly to the context in which these data occurred in the data set used to develop DASNG neural network.
- **S – semantic** – means that all semantically related objects are directly or indirectly connected to enable fast access to them if necessary.
- **N – neural** – because a special associative model of spiking neurons is used to represent attribute data, their ranges or subsets, as well as objects, clusters, classes etc.
- **G – graph** – because all neurons are connected in a sparse graph structure that represents associations between data and objects.

DASNG Features

- **DASNG** contains all horizontal relations between objects that are implemented in relational model.
- **DASNG** naturally implements many vertical relations beteen objects thanks to aggregations of duplicates and connections between neurons representing similar (ordered) attribute values.
- **DASNG** always puts new data into the context of other stored data.
- **DASNG** use an associative spiking neurons to implement reactive functionality and automatic inference according to the initial context.
- **DASNG** significantly decreases computational complexity for many operations because it replaces complex operations by its structure.
- **DASNG** replaces many time consuming loops on tabular structures.

DEF: We say that the structure replaces operations performed on another data structure when the computational complexity of the operations on that structure decreases to constant computational complexity O(1).

DASNG construction for DB

Transforms only these tables for which all foreign keys are already represented by the neurons in the DASNG:

Patients

Social Security No	First Name	Last Name	Blood Group	Age	ID Room
17102595834	Jack	Hanks	0	0	R6
17102785603	Nina	Rock	AB	0	R6
73040593020	Tom	Kite	A	44	R4
84021205010	Tom	Ford	AB	33	R1
90503029943	Emy	Cruise	A	27	R2
94051382054	Lisa	White	B	23	R3
94080639502	Paula	Smith	B	23	R2

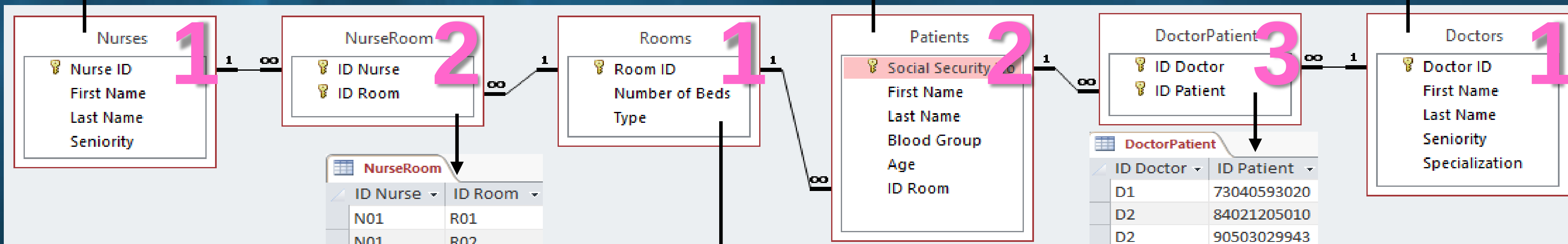
Nurses

Nurse ID	First Name	Last Name	Seniority
N1	Amy	Moon	12
N2	Rose	Jolie	18
N3	Kate	Ford	24
N4	Lisa	Brown	9
N5	Sara	Pitt	4
N6	Kate	Lopez	12

Transformation Process

Doctors

Doctor ID	First Name	Last Name	Seniority	Specialization
D1	Tom	Hanks	18	orthopedics
D2	Jack	Brown	15	surgery
D3	Lisa	Ford	23	pediatrician
D4	Tom	Trump	35	pediatrician
D5	Kate	Smith	7	surgery
D6	Amy	Hanks	12	surgery



NurseRoom

ID Nurse	ID Room
N01	R01
N01	R02
N02	R06
N03	R03
N03	R04
N03	R05
N04	R01
N04	R02
N05	R03
N05	R04
N05	R05
N06	R06

DoctorPatient

ID Doctor	ID Patient
D1	73040593020
D2	84021205010
D2	90503029943
D3	17102595834
D3	17102785603
D4	17102595834
D4	17102785603
D5	90503029943
D5	94080639502
D6	17102595834
D6	73040593020
D6	94051382054
D6	84021205010
D6	94080639502

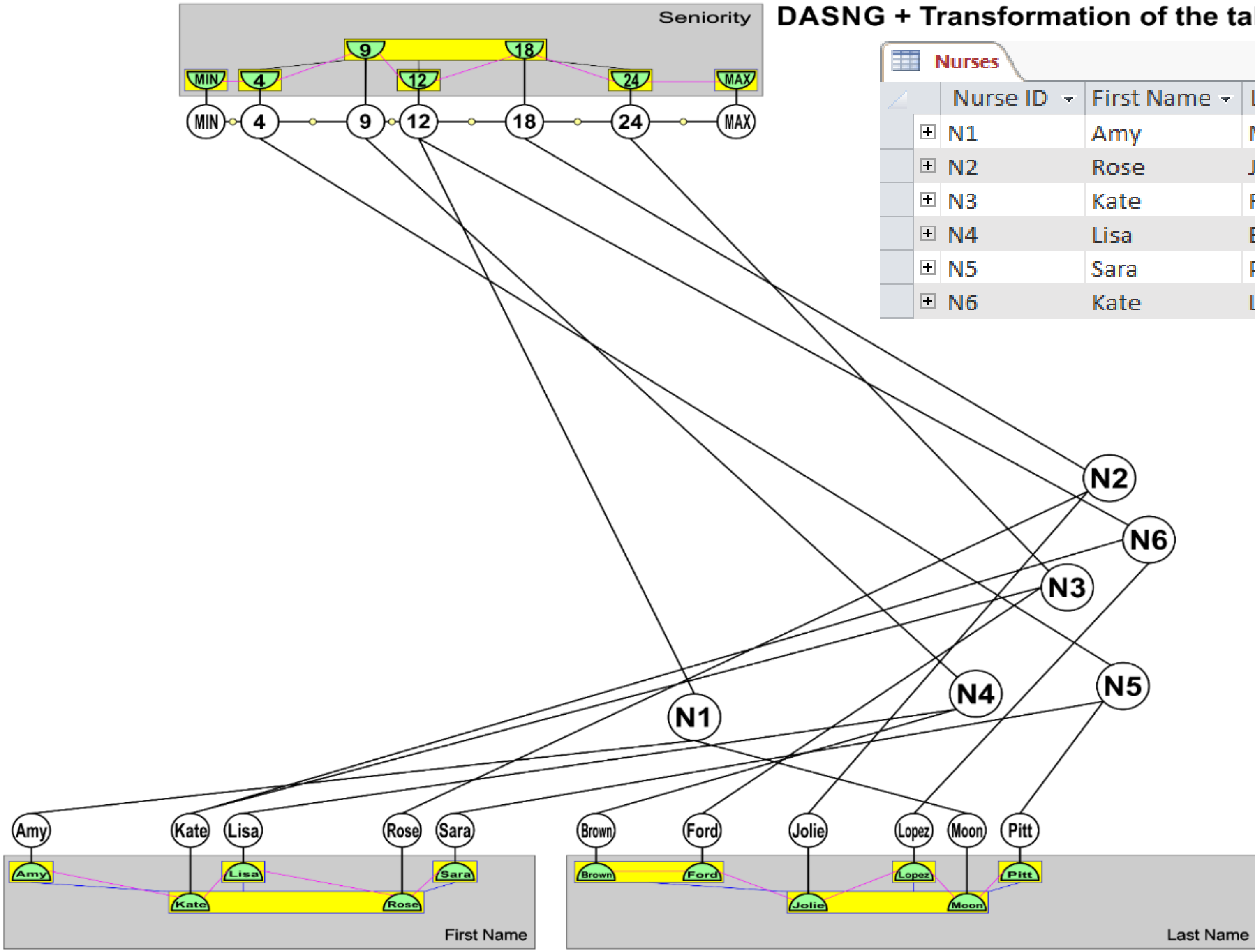
Rooms

Room ID	Number of Beds	Type
R1	2	intensive care
R2	2	intensive care
R3	3	therapeutic
R4	3	therapeutic
R5	3	therapeutic
R6	8	newborns

Possible sequence of transformation of tables

DASNG construction for DB

The table NURSES is added to the empty DASNG network.



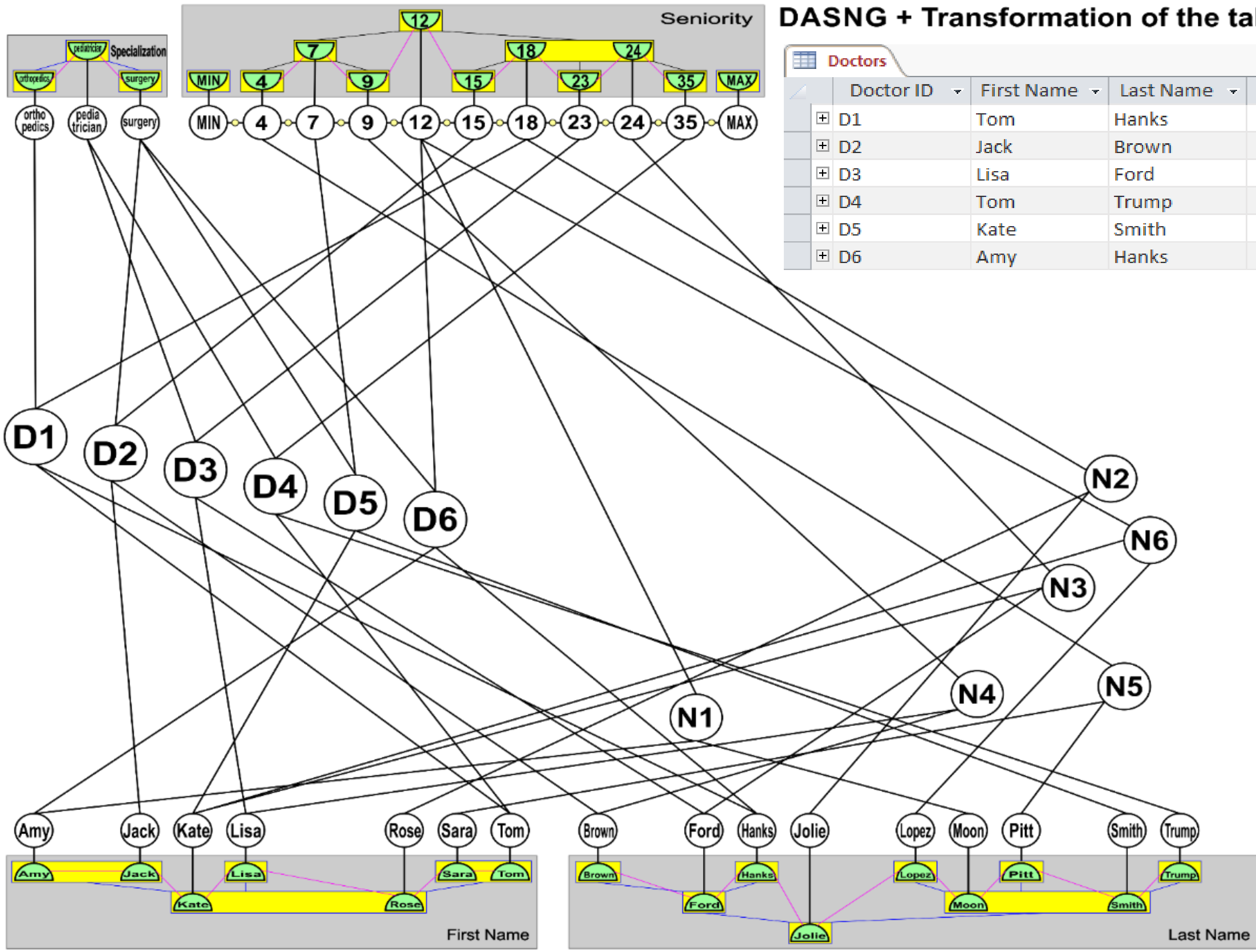
DASNG + Transformation of the table NURSES

Nurses

Nurse ID	First Name	Last Name	Seniority
N1	Amy	Moon	12
N2	Rose	Jolie	18
N3	Kate	Ford	24
N4	Lisa	Brown	9
N5	Sara	Pitt	4
N6	Kate	Lopez	12

DASNG construction for DB

The table DOCTORS is added to the DASNG network.



DASNG + Transformation of the table DOCTORS

Doctors

Doctor ID	First Name	Last Name	Seniority	Specialization
D1	Tom	Hanks	18	orthopedics
D2	Jack	Brown	15	surgery
D3	Lisa	Ford	23	pediatrician
D4	Tom	Trump	35	pediatrician
D5	Kate	Smith	7	surgery
D6	Amy	Hanks	12	surgery

DASNG construction for DB

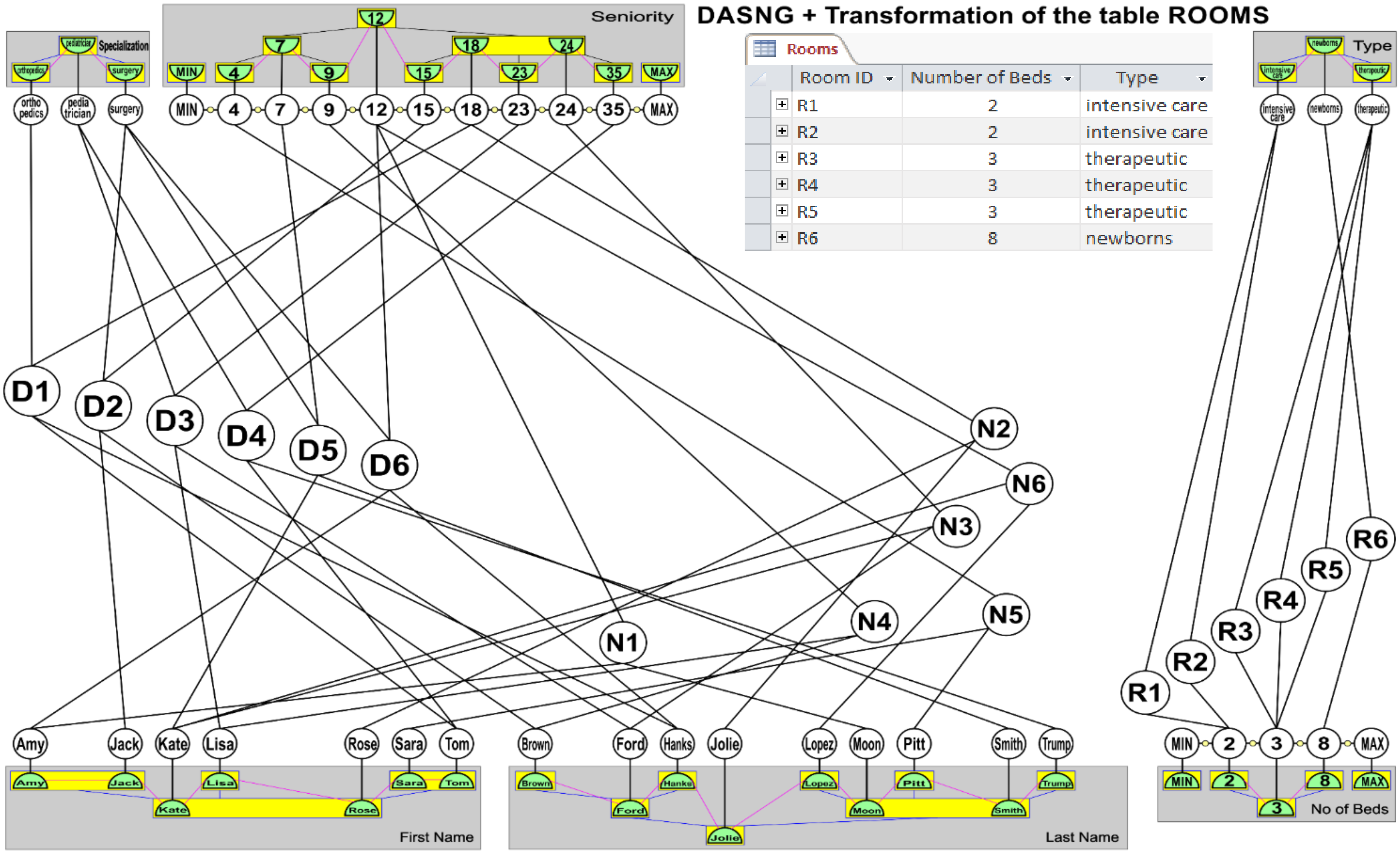
AGH

The table ROOMS is added to the DASNG network.

DASNG + Transformation of the table ROOMS

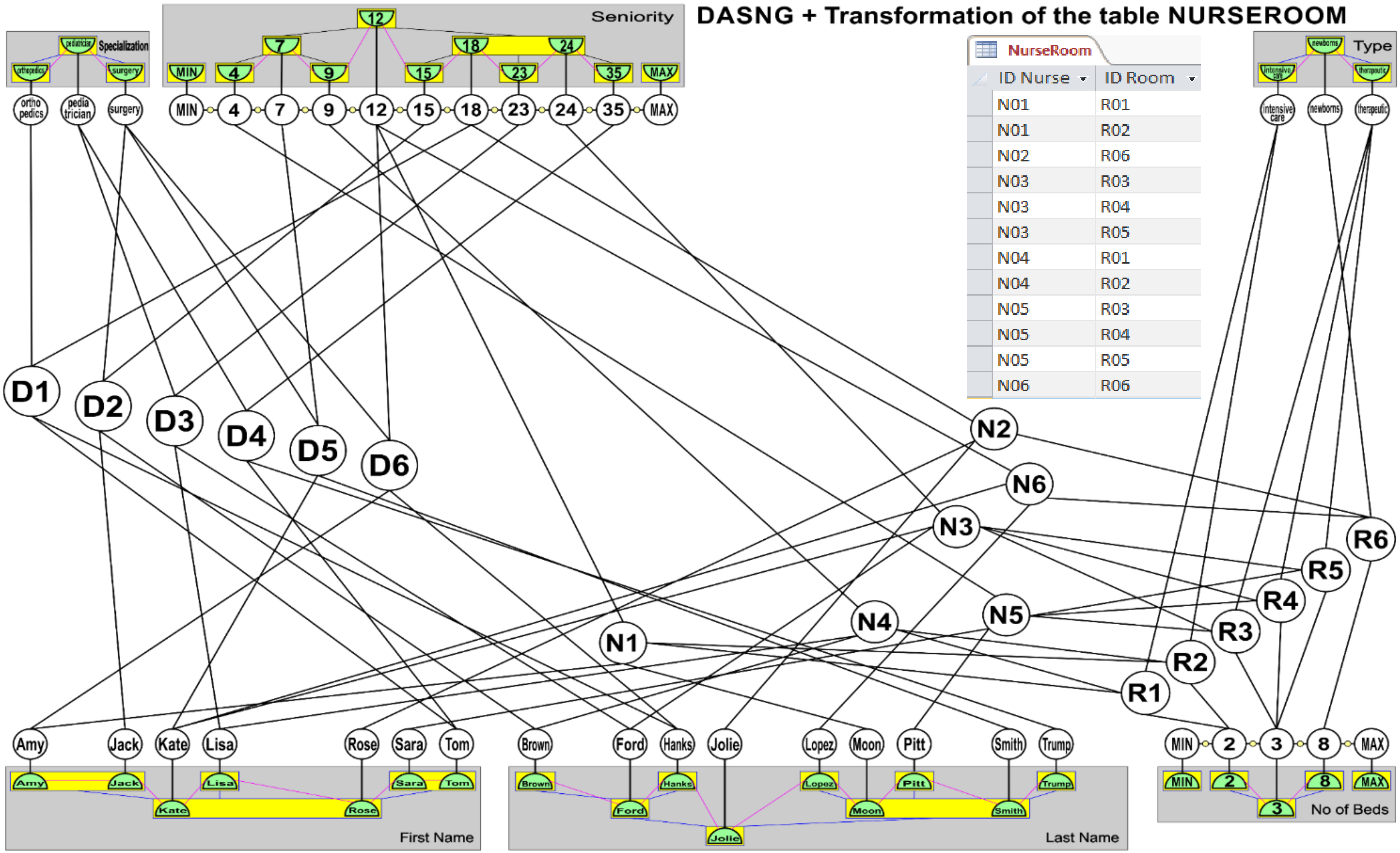
Rooms

Room ID	Number of Beds	Type
R1	2	intensive care
R2	2	intensive care
R3	3	therapeutic
R4	3	therapeutic
R5	3	therapeutic
R6	8	newborns



DASNG construction for DB

The table NURSEROOM is added to the DASNG network.

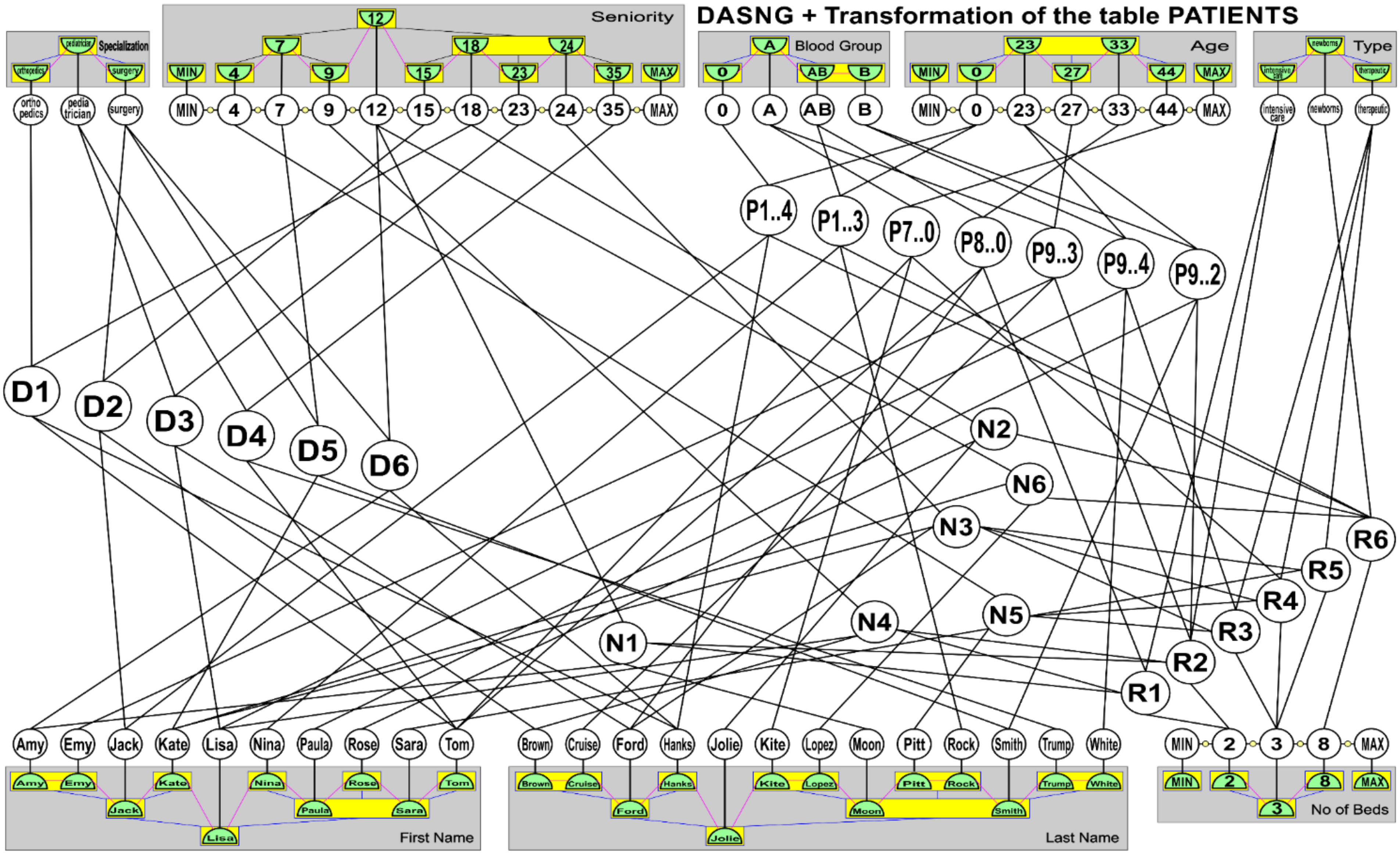


NurseRoom

ID Nurse	ID Room
N01	R01
N01	R02
N02	R06
N03	R03
N03	R04
N03	R05
N04	R01
N04	R02
N05	R03
N05	R04
N05	R05
N06	R06

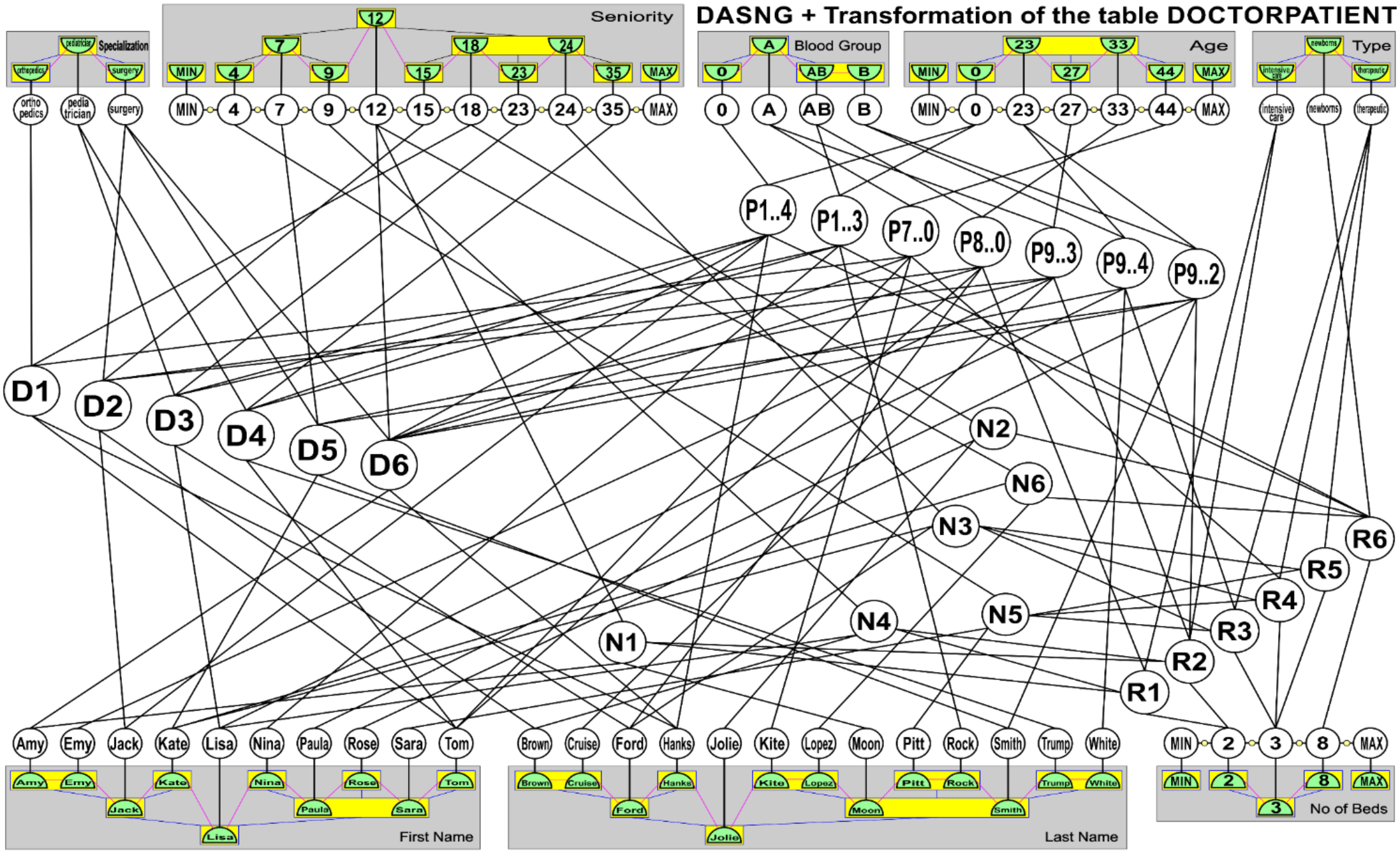
DASNG construction for DB

The table PATIENTS is added to the DASNG network.



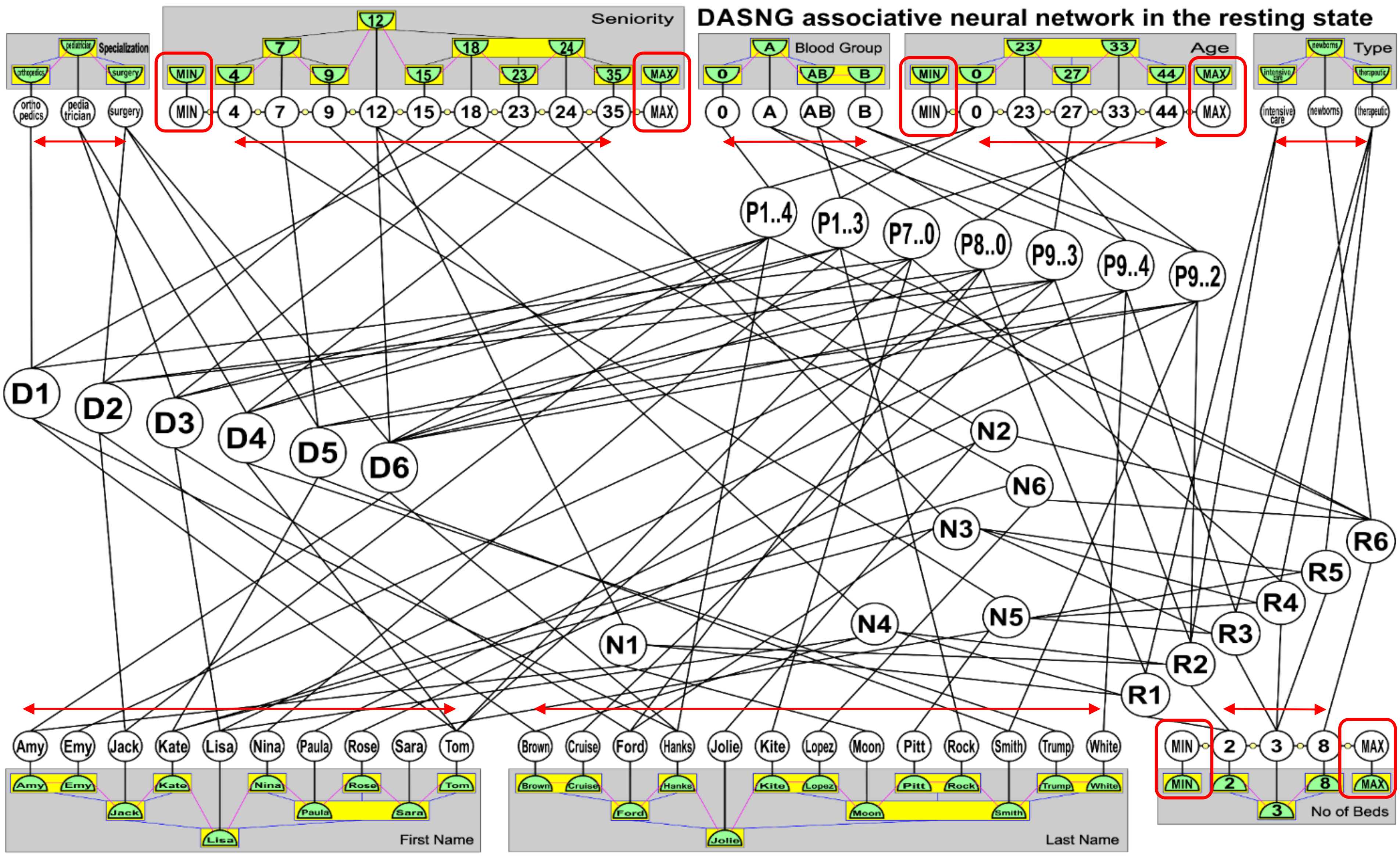
DASNG construction for DB

The table DOCTORPATIENT is added to the DASNG network.



Result of the associative transformation of the DB to the DASNG network:

No duplicates and all values are sorted and quickly accessible!



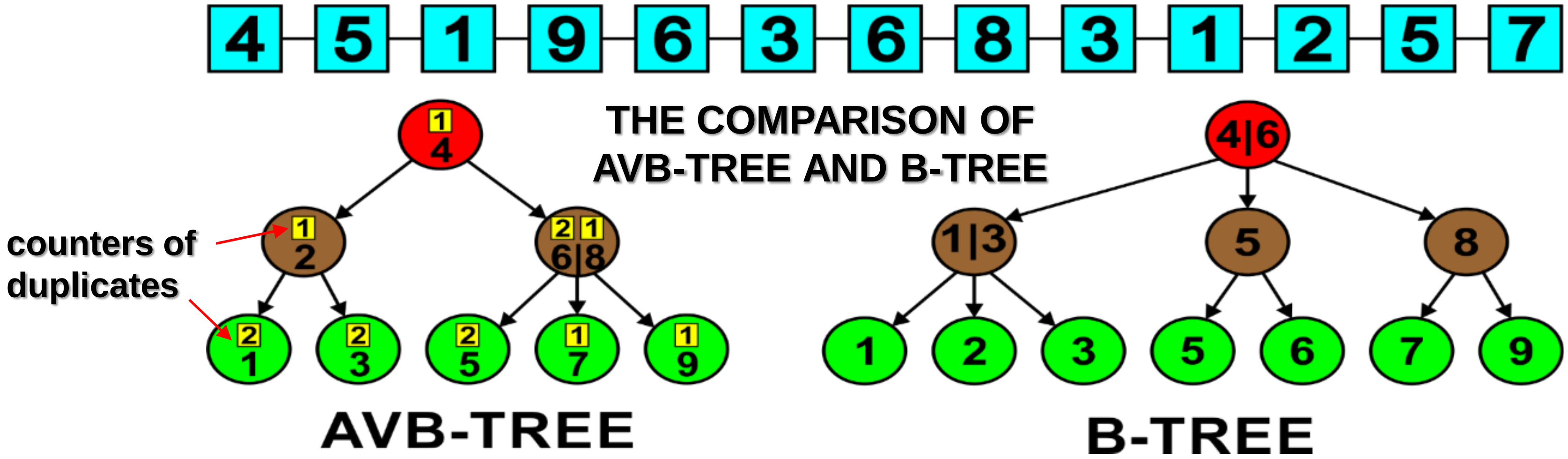
DASNG uses AVB-trees for fast attribute data access

AVB-tree is a new self-ordering and self-balancing tree structure that enables to efficiently organize attribute values and achieve very fast access to all stored feature values and objects in the DASNG network.

AVB-trees are very similar to **B-trees** but AVB-trees additionally aggregate and count up all duplicated values.

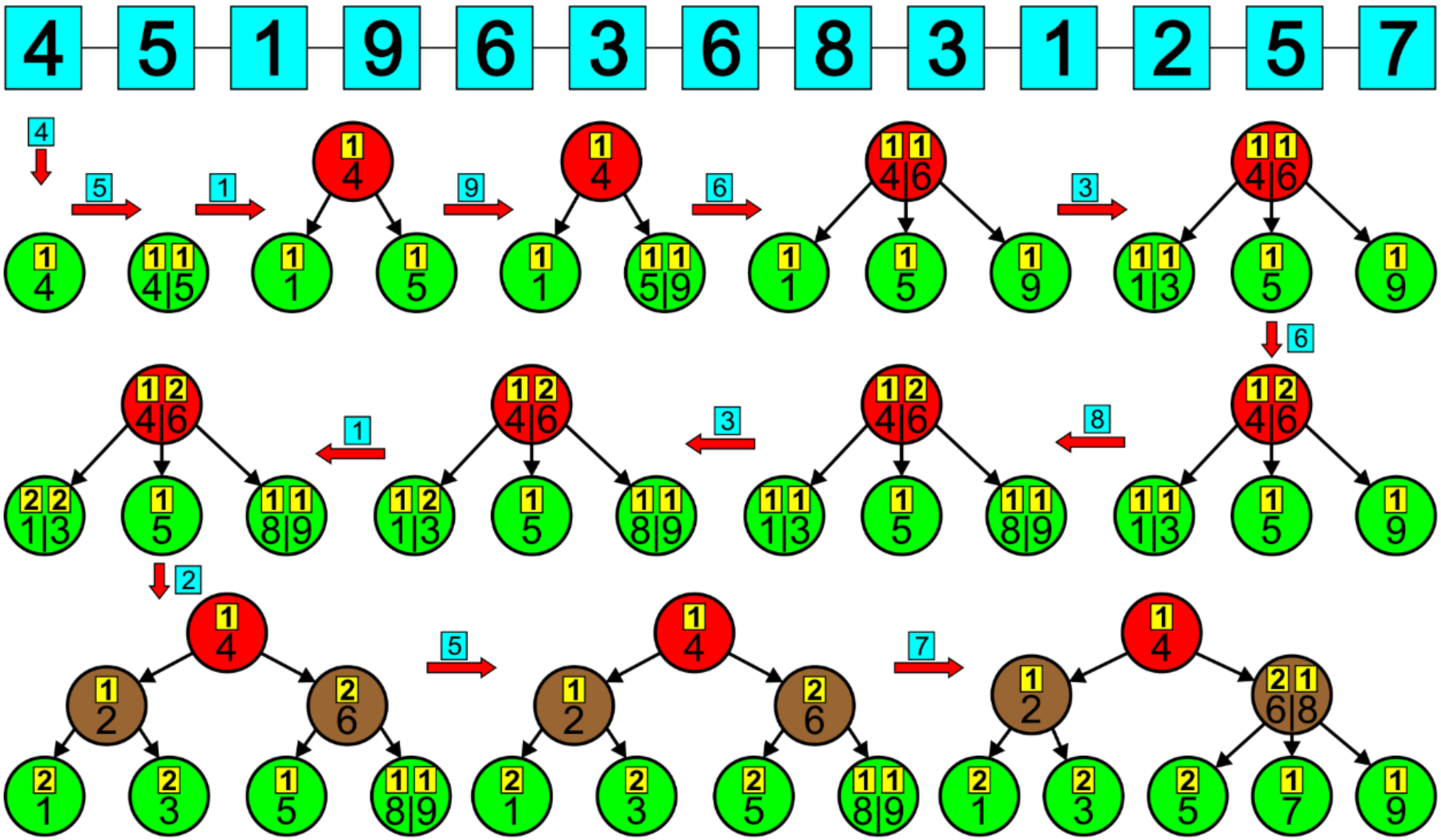
The aggregations of duplicates result in typically much smaller number of nodes of **AVB-trees** than achieved for **B-trees** for the same collection of data.

Search operations are usually also **faster** taking usually **constant time**!



AVB-trees construction

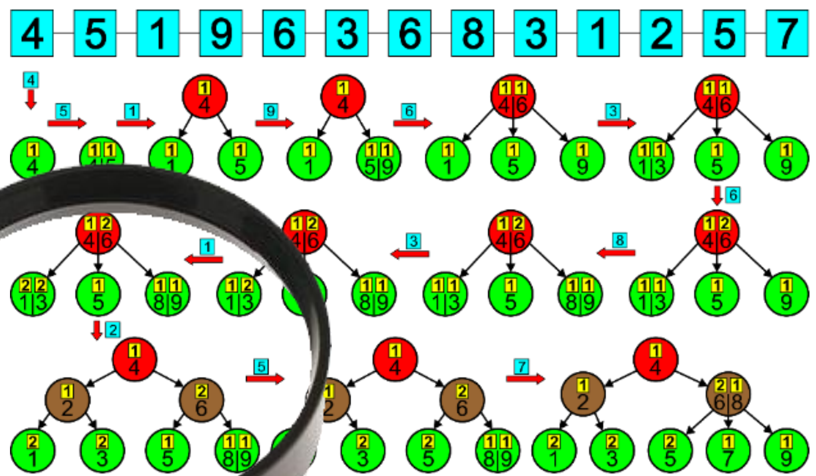
AVB-trees are constructed similarly to B-trees but duplicates are aggregated (represented once) and counted up:

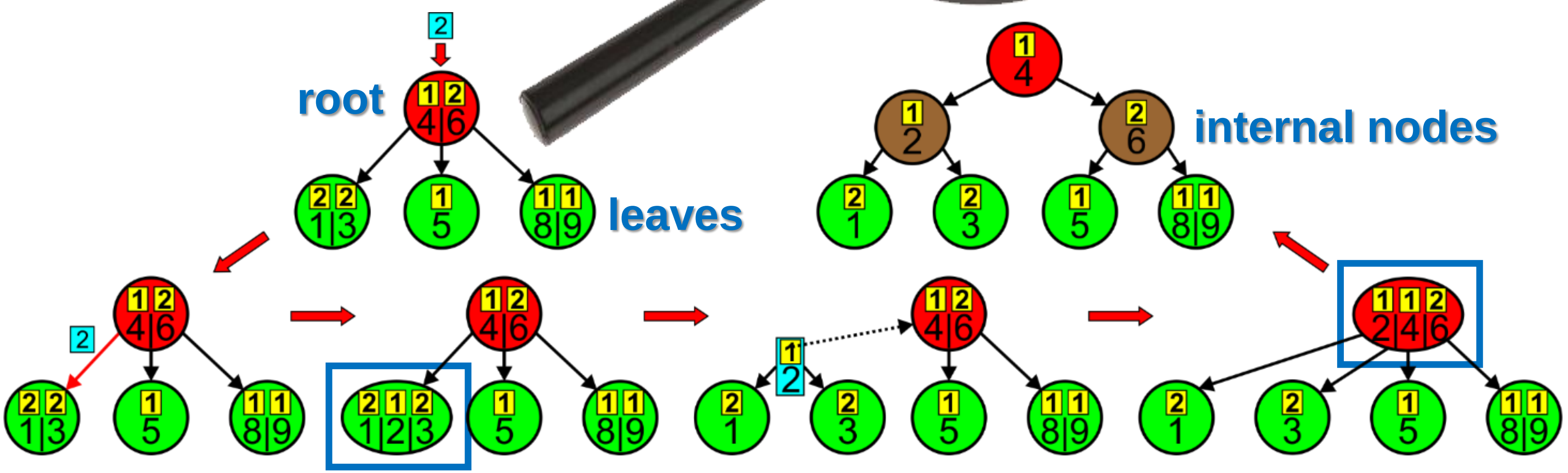


AVB-trees construction

AVB-trees are fully self-balancing:

When the node contains more than two keys (values or sensors) it is automatically divided as shown in this sample presenting the intermediate operations on this AVB-tree:

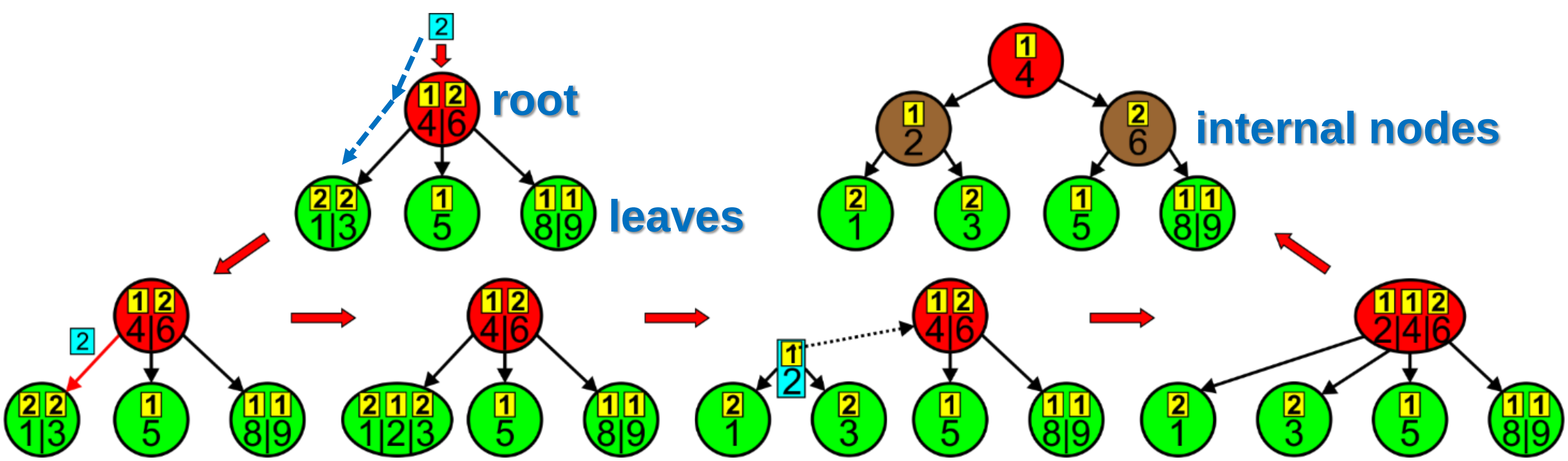




AVB-trees construction algorithm: INSERT operation

1. Start **from the root** and **go recursively down** along the edges to the descendants **until the leaf** is not achieved after the following rules:

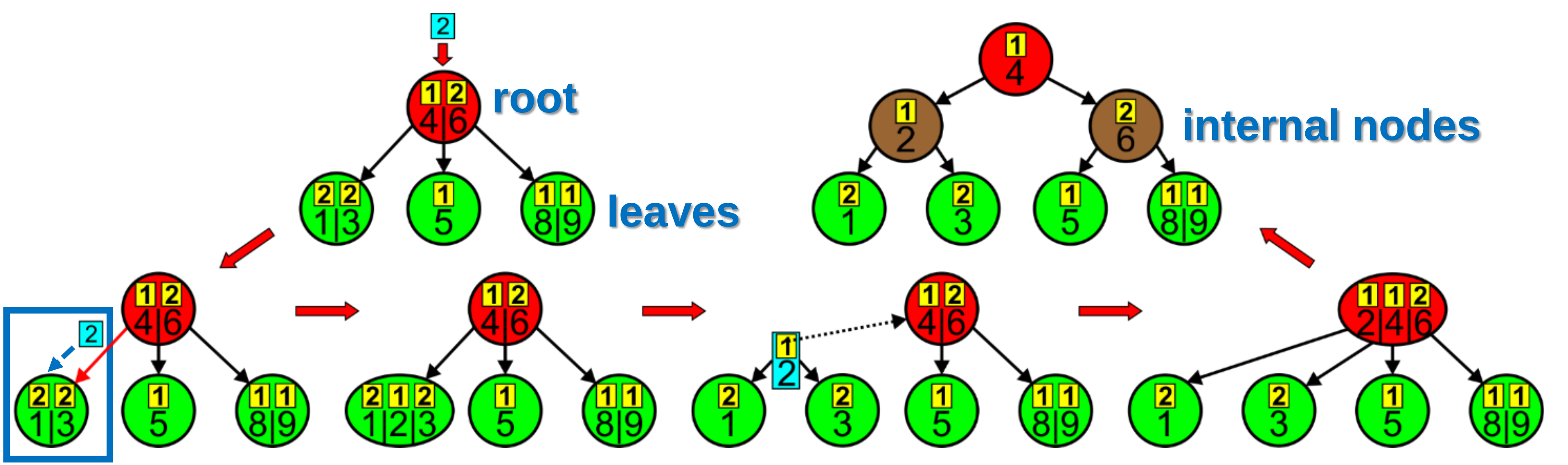
- if one of the keys stored in the node **equals** to the inserted key, **increment** the counter of this key, and finish this operation,
- else **go to the left child** node if the inserted key is **less than** the leftmost key in the node,
- else **go to the right child** node if the inserted key is **greater than** the rightmost key in the node,
- else **go to the middle child** node.



AVB-trees construction algorithm: INSERT operation

2. When the **leaf** is achieved:

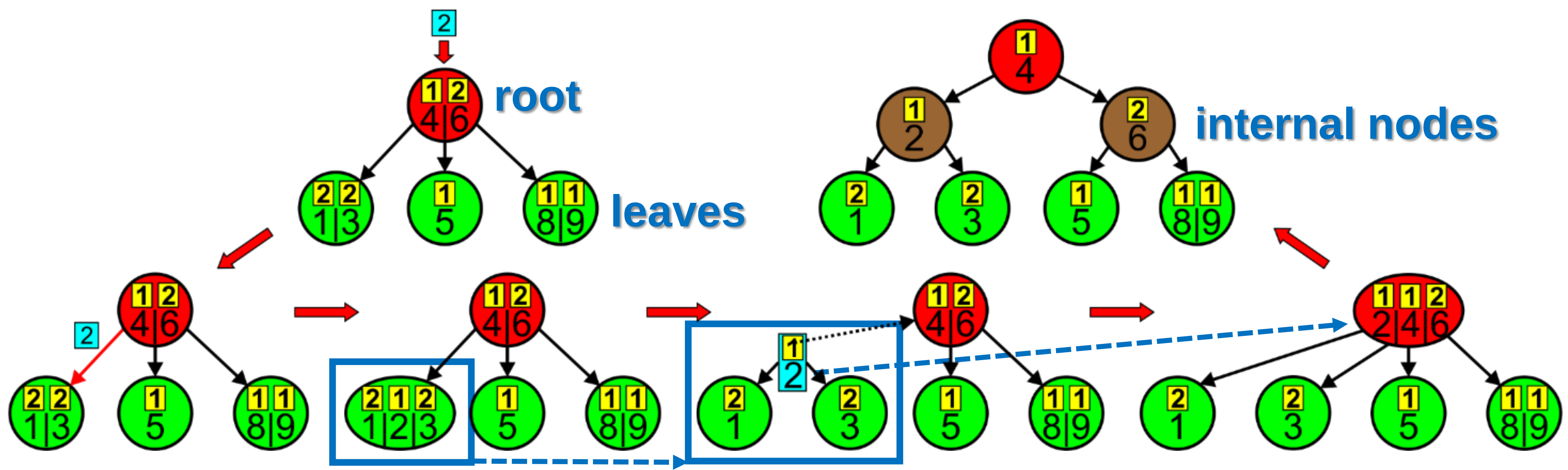
- and if the inserted key is **equal** to one of the keys in this leaf, **increment** the counter of this key, and finish this operation,
- else **insert** the inserted key to the keys stored in this leaf **in the increasing order**, initialize its counter to one, and go to step 3.



AVB-trees construction algorithm: INSERT operation

3. If the number of all keys stored in this leaf is greater than two, divide this leaf into two leaves in the following way:

- let the divided leaf represent the leftmost (least) key together with its counter;
- create a new leaf and let it to represent the rightmost (greatest) key together with its counter;
- and the middle key together with its counter and the pointer to the new leaf representing the rightmost key pass to the parent node if it exists, and go to step 4;
- if the parent node does not exist, create it (a new root of the AVB-tree) and let it represent this middle key together with its counter, and create new edges to the divided leaf representing the leftmost key and to the leaf pointed by the passed pointer to the new leaf representing the rightmost key. Next, finish this operation.

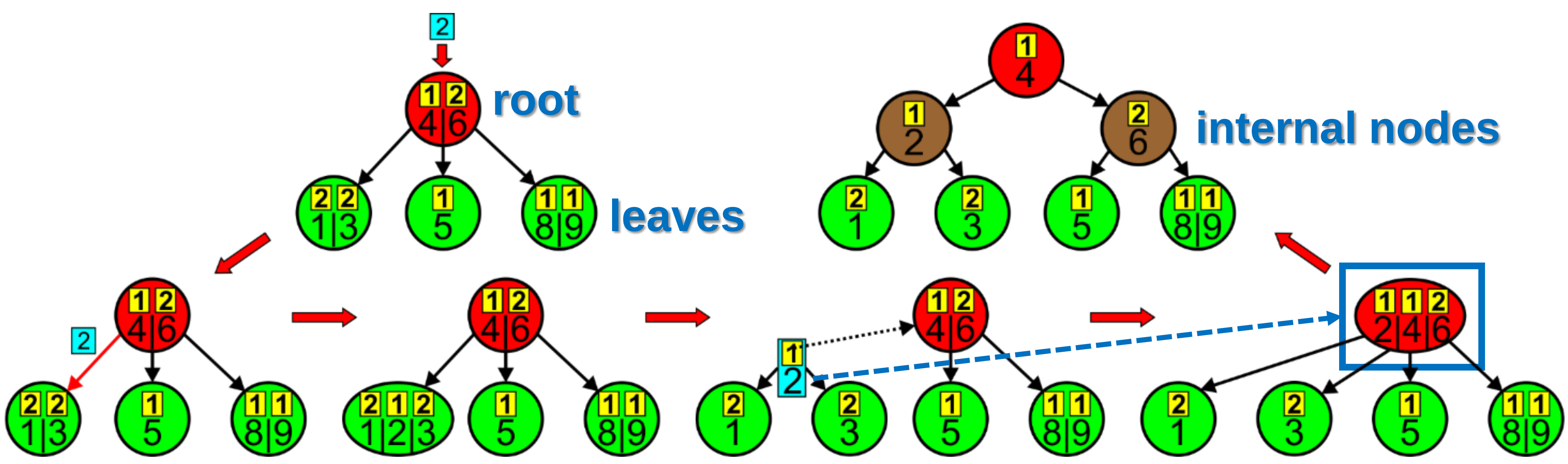


AVB-trees construction algorithm: INSERT operation

4. **Insert** the passed key together with its counter to the key(s) stored in this node **in the increasing order** after the following rules:

- if the key comes from the left branch, insert it on the left side of the key(s);
- if the key comes from the right branch, insert it on the right side of the key(s);
- if the key comes from the middle branch, insert it between the existing keys.

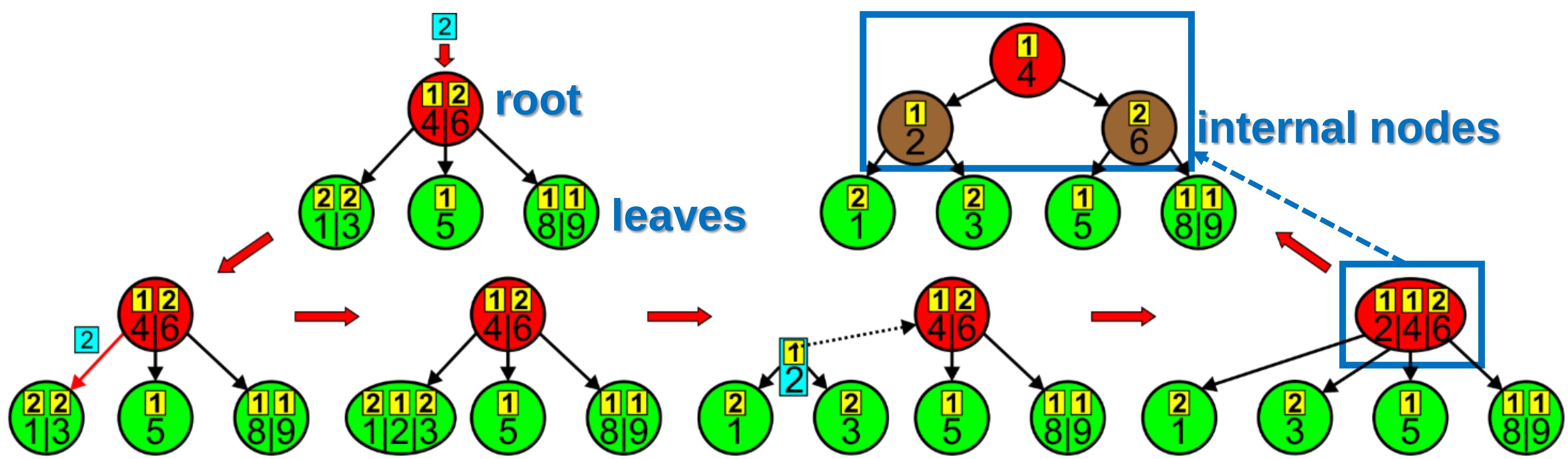
5. **Create a new edge** to the new leaf or node pointed by the passed pointer and **insert this pointer** to the child list of pointers immediately after the pointer representing the edge to the divided leaf or node.

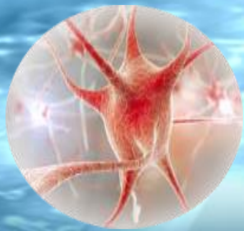


AVB-trees construction algorithm: INSERT operation

6. If the number of all keys stored in this node is greater than two, divide this node into two nodes in the following way:

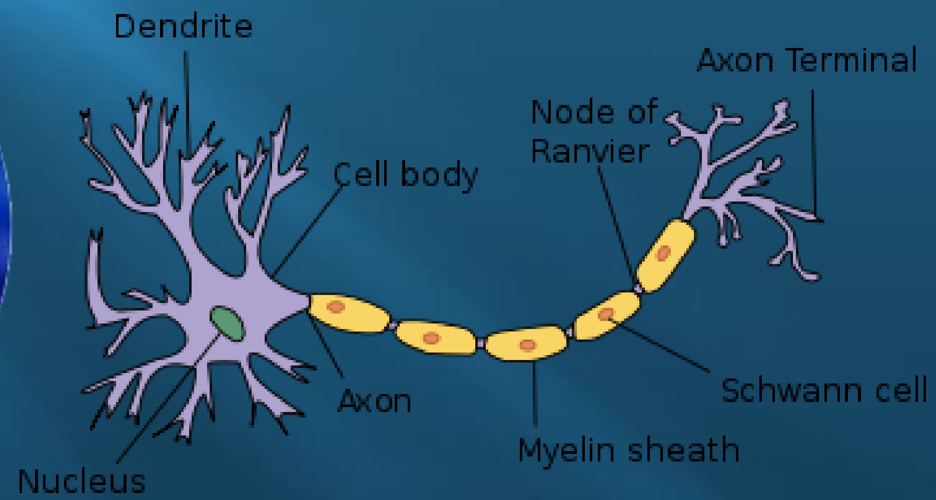
- let the existing node represent the leftmost (least) key together with its counter;
- create a new node and let it represent the rightmost (greatest) key together with its counter;
- the middle key together with its counter and the pointer to the new node representing the rightmost key pass to the parent node if it exists and go back to step 4;
- if the parent node does not exist, create it (a new root of the AVB tree), let it represent this middle key together with its counter, and create new edges to the divided node representing the leftmost key and to the node pointed by the passed pointer to the new node representing the rightmost key. Next, finish this operation.

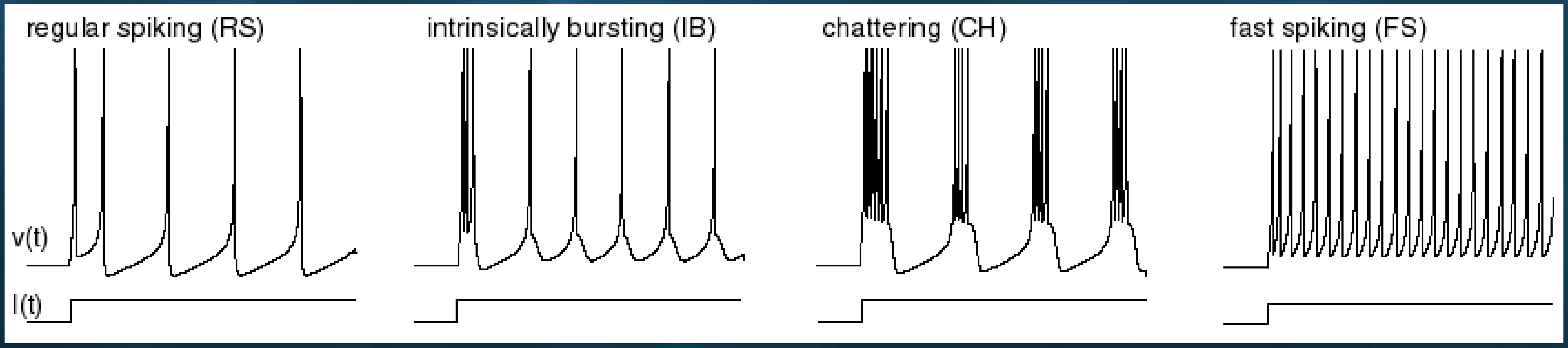


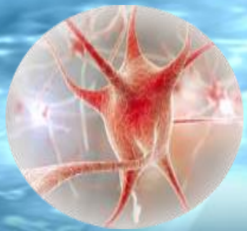


DASNG uses Sensors and Associative Spiking Neurons

Sparsely and contextually connected neural networks play important role in the associative processes in the brain where knowledge is represented. The DASNG uses models of neurons which incorporate the concept of time.







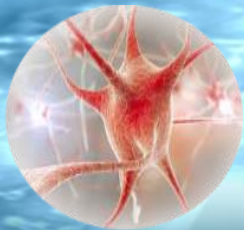
Fundamental problem of neuron communication

The fundamental problem is to propose the model that explains **how information is encoded and decoded** by a series of pulses, i.e. action potentials?!

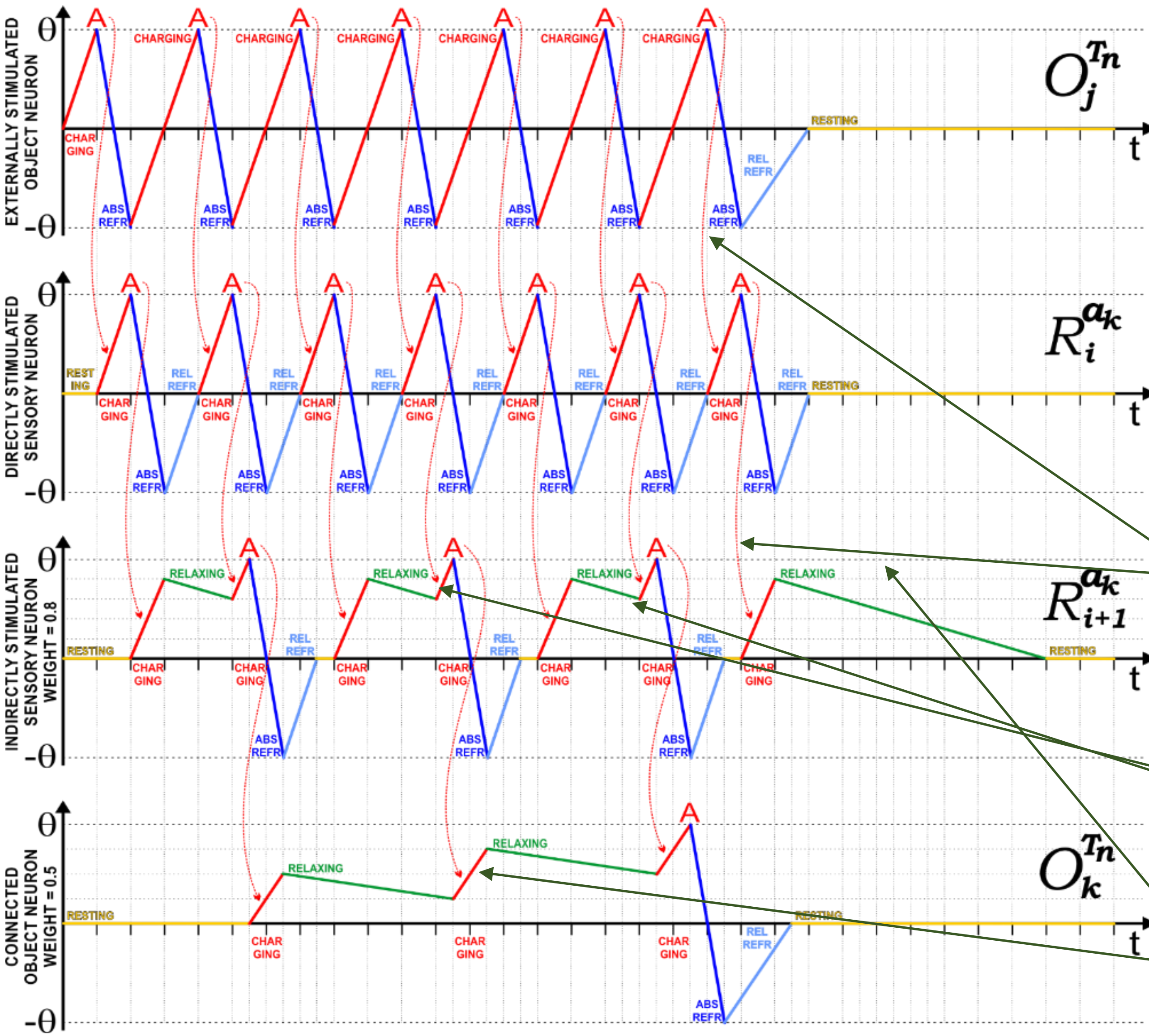
The **fundamental question of neuroscience** is to determine **whether neurons communicate by a rate or temporal code?**

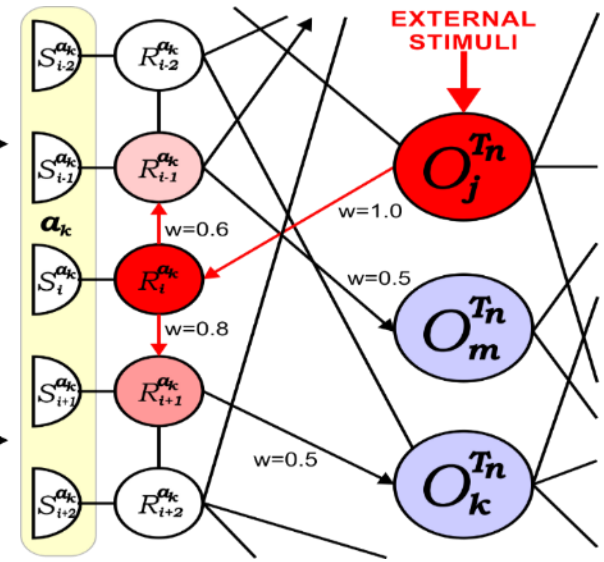
Temporal coding suggests that a single spiking neuron can replace hundreds of hidden units on a sigmoidal neural network. Is that true?

Experiments on DASNG networks revealed that **both time and rate** have an appropriate influence on postsynaptic neuron activity, and thus on what this neuron represents.

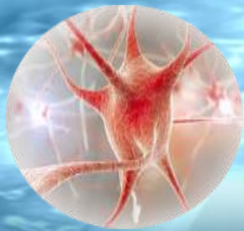


How do Associative Spiking Neurons work and Influence other Neurons?

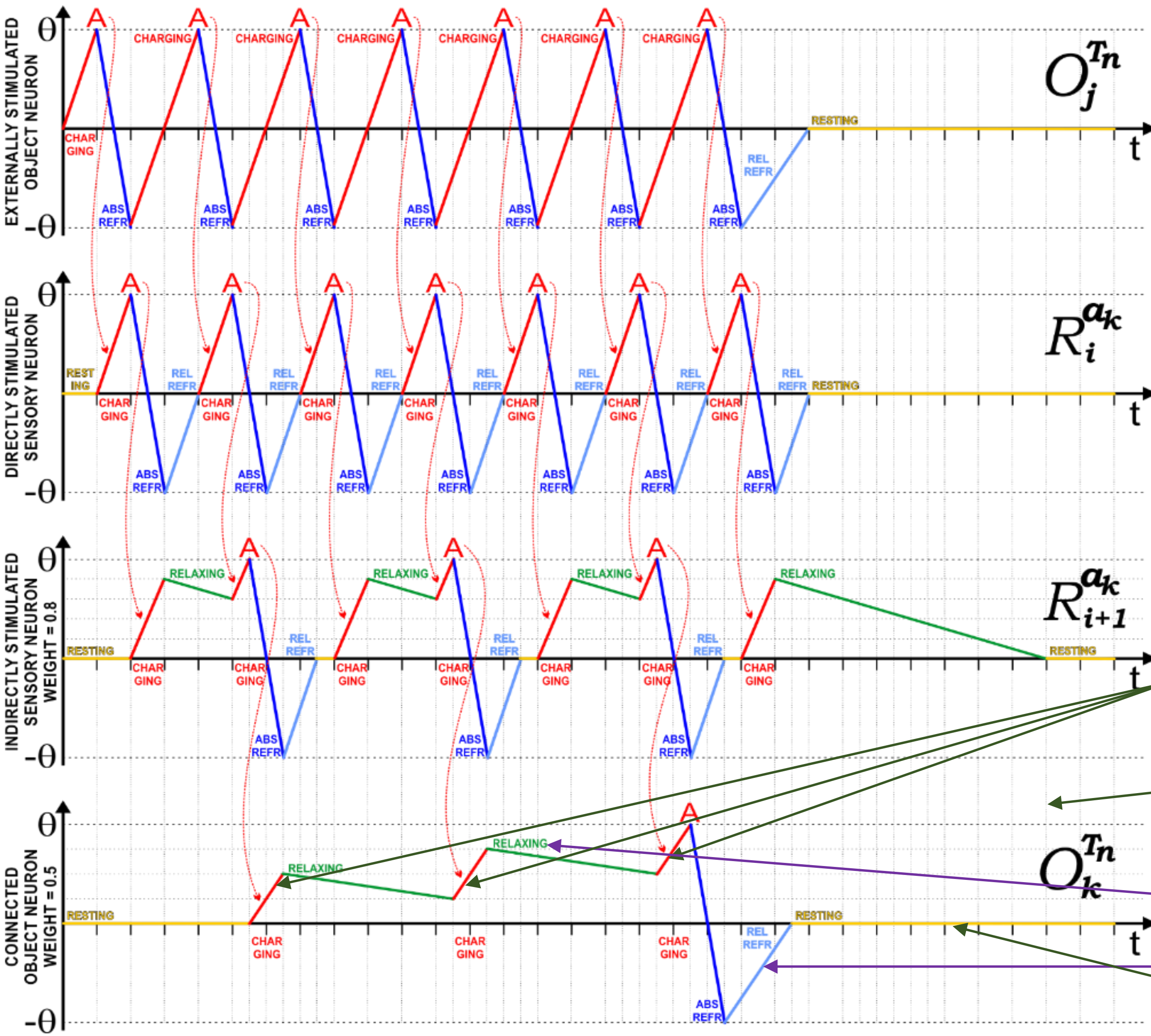


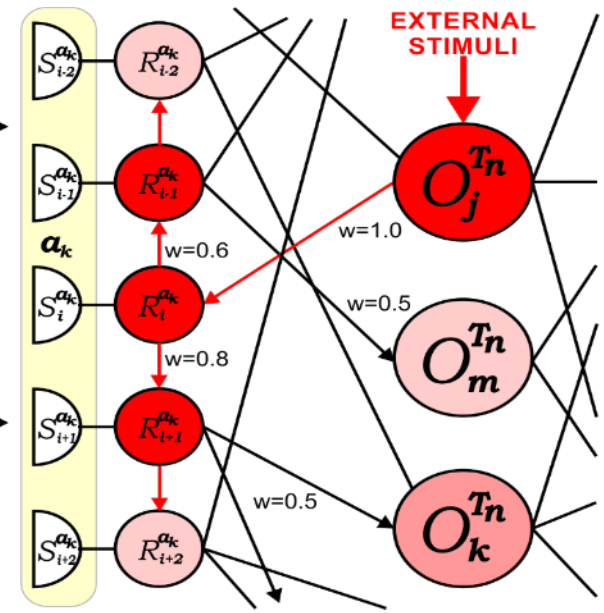


Each activation of the neuron O_j **stimulates and activates** the neuron R_i which **stimulates** the neighboring sensory neurons R_{i+1} and R_{i-1} with the force equal to the weights of these connections, i.e. 0.8 and 0.6, appropriately. It is therefore necessary to **stimulate these neurons twice**, so that, with regards to **relaxation**, they achieve a total stimulus greater than their **activation thresholds** $\theta = 1$. This will allow them for activation and then to start **stimulation** of the connected neurons, e.g. the neuron O_k.



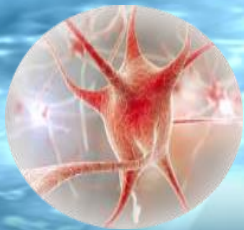
How do Associative Spiking Neurons work and Influence other Neurons?



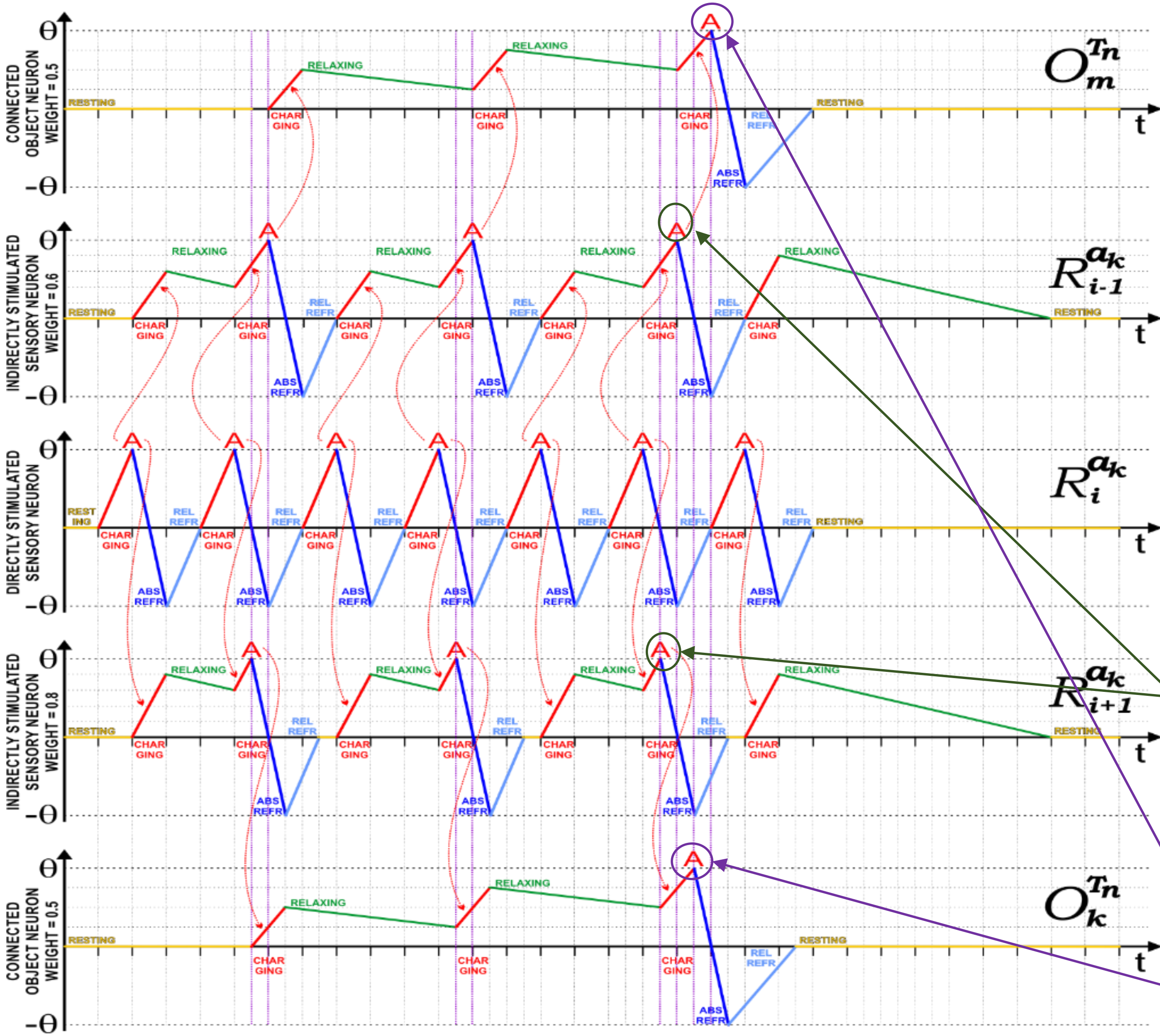


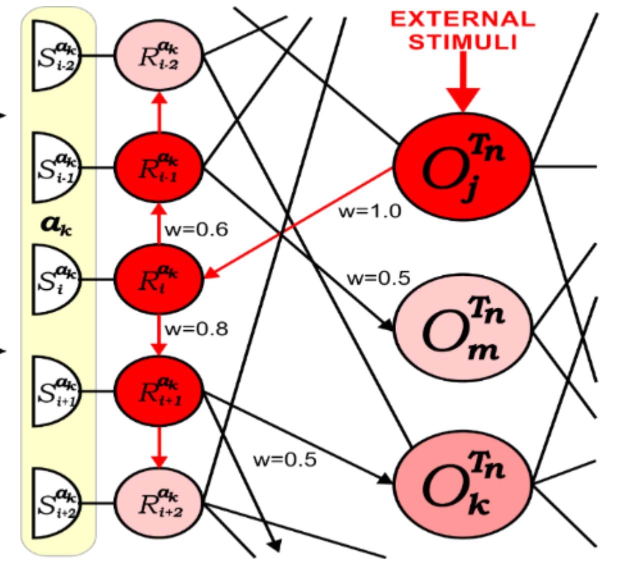
As we can notice, the neuron O_k needs to be **stimulated triple times** through the connection coming from the neuron R_{i+1} and weighted with 0.5 to reach the **activation threshold** θ = 1.0.

When a neuron is not externally stimulated, the **relaxation** and **refraction** processes try to restore the **resting state** in it.



How do Associative Spiking Neurons work and Influence other Neurons?





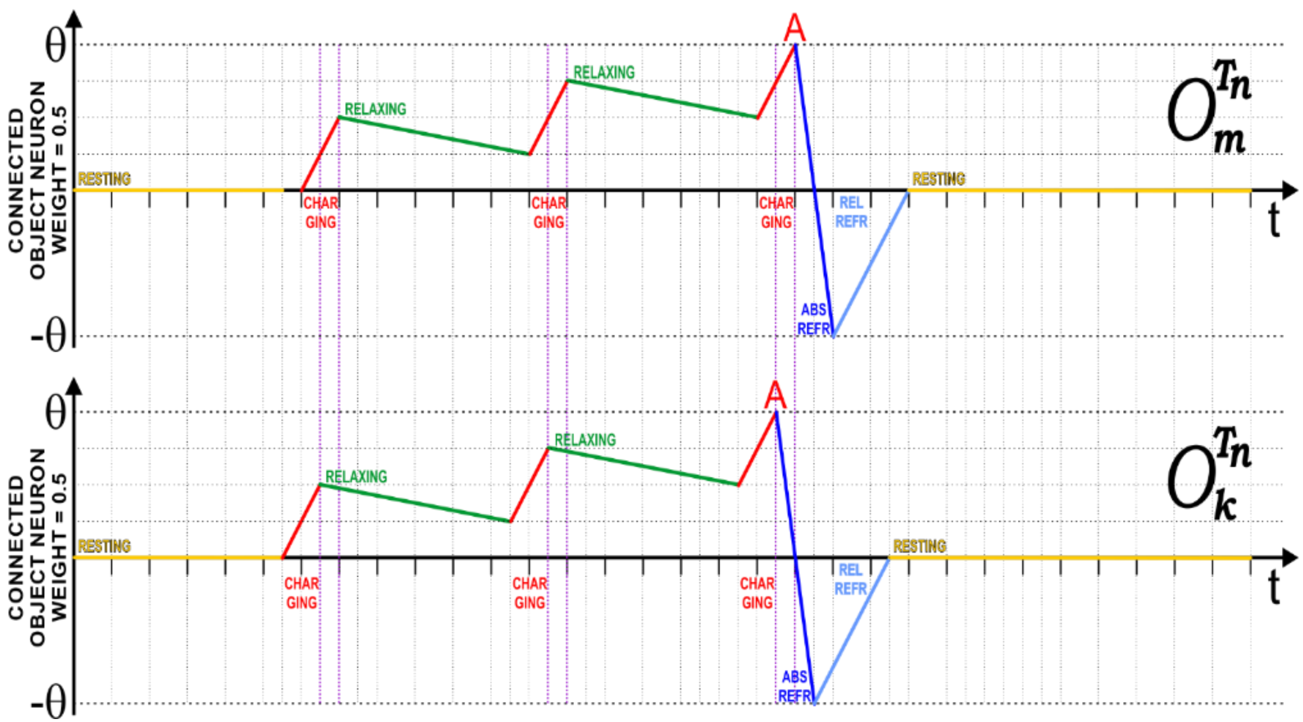
The sensory neurons R_{i+1} and R_{i-1} are stimulated with different strength according to the weights (0.8 and 0.6) of connections coming from the neuron R_i. It induces different **excitation levels** inside them and **different activation moments**. The neuron R_{i+1} achieves this threshold **earlier** than the neuron R_{i-1}, so the neuron R_{i+1} starts **earlier** to stimulate the neuron O_k than the neuron R_{i-1} starts to influence the neuron O_m. Thus, the neuron O_k will be activated **earlier** than the neuron O_m. It implies **greater similarity** of the object represented **by the neuron O_k** than **by the neuron O_m**. This is consistent with **intuition** of real **similarity**.

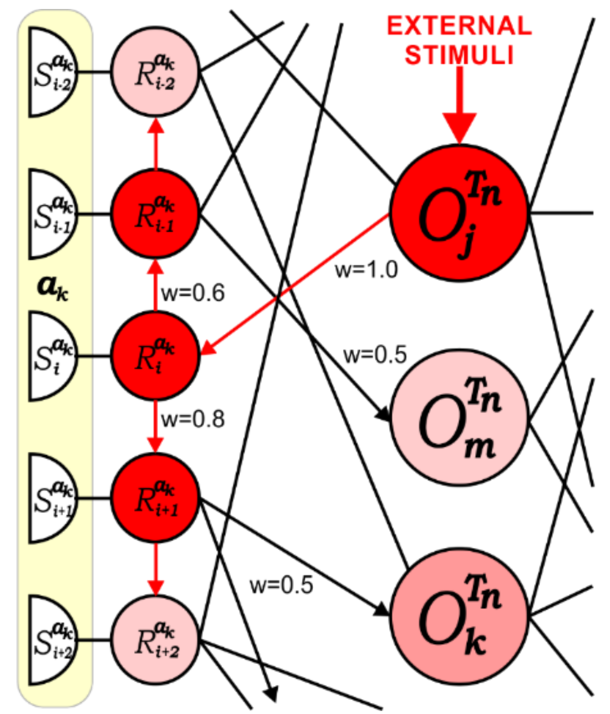
How do Associative Spiking Neurons work and Influence other Neurons?

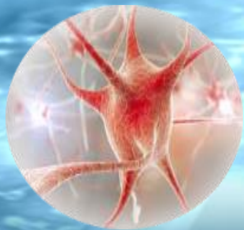
The **small shift in activation** of the neurons O_k and O_m may seem to be insignificant or negligible, but this phenomenon is crucial for the working way of **biological neural networks** as well as of the introduced **associative neural graphs DASNG**.

The difference in activation time of these neurons representing different objects informs us of **weaker or stronger associations** with these objects, i.e. **less or greater similarity** of them.

In this way, **associative spiking neurons** automatically **conclude**, revealing their various relationships with other objects and data represented by other connected neurons.







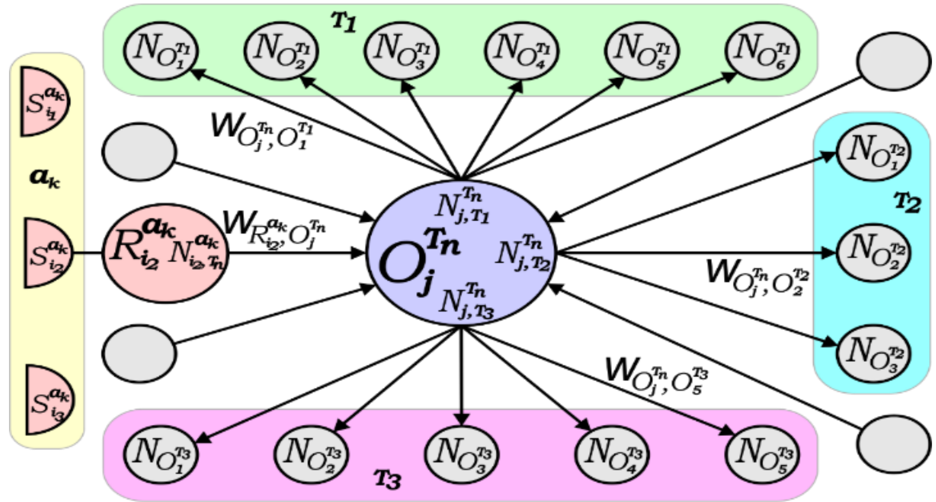
Connection Weights

Orderable sensory neurons are connected,
the connections are weighed expressing similarity:

$$w_{R_{v_i}^{a_k}, R_{v_j}^{a_k}} = 1 - \frac{\left| v_i^{a_k} - v_j^{a_k} \right|}{r^{a_k}}$$

The connections between the sensory and object neurons are **weighted** in the following way:

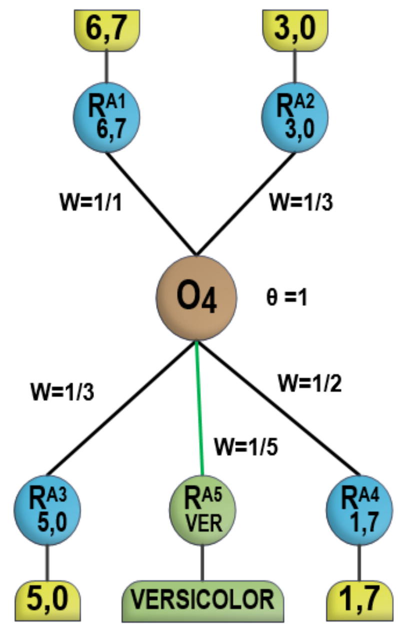
$$w_{R_{v_i}^{a_k}, O_j^{T_n}} = \frac{1}{\left\| v_i^{a_k} \right\|} \qquad w_{O_j^{T_n}, R_{v_i}^{a_k}} = \theta_{R_{v_i}^{a_k}} = 1$$

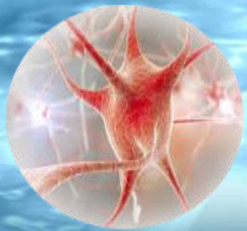


The weights of synaptic connections between various object neurons are computed on the basis of the number of objects represented by the object neurons of the considered layer of the DASNG, which represents a single database table. If the given **object neuron of the considered layer** is connected to **M object neurons of another layer**, then the weight is computed in the following way:

$$w_{O_j^{T_n}, O_k^{T_m}} = \frac{1}{N_{j,T_m}^{T_n}} \cong \frac{1}{M} \qquad w_{O_k^{T_m}, O_j^{T_n}} = \frac{1}{N_{k,T_n}^{T_m}} \cong \frac{1}{N}$$

where $N_{k,T_n}^{T_m} = N = 1$ for the relations one-to-many (1:M) and the relations many-to-many (N:M). The equation is precise when there are no duplicates of the whole records in the database. We need to create separate lists of connections in each neuron to represent connections to neurons of various layers in order to easily compute the number of objects $N_{j,T_m}^{T_n}$ or the number of connections M.





Activation Thresholds

AGH

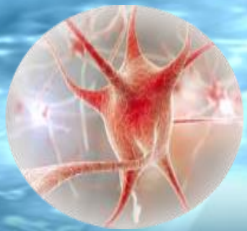
Activation thresholds of sensory neurons:

$$\theta_{R_{v_i}^{a_k}} = 1$$

Activation thresholds of object neurons:

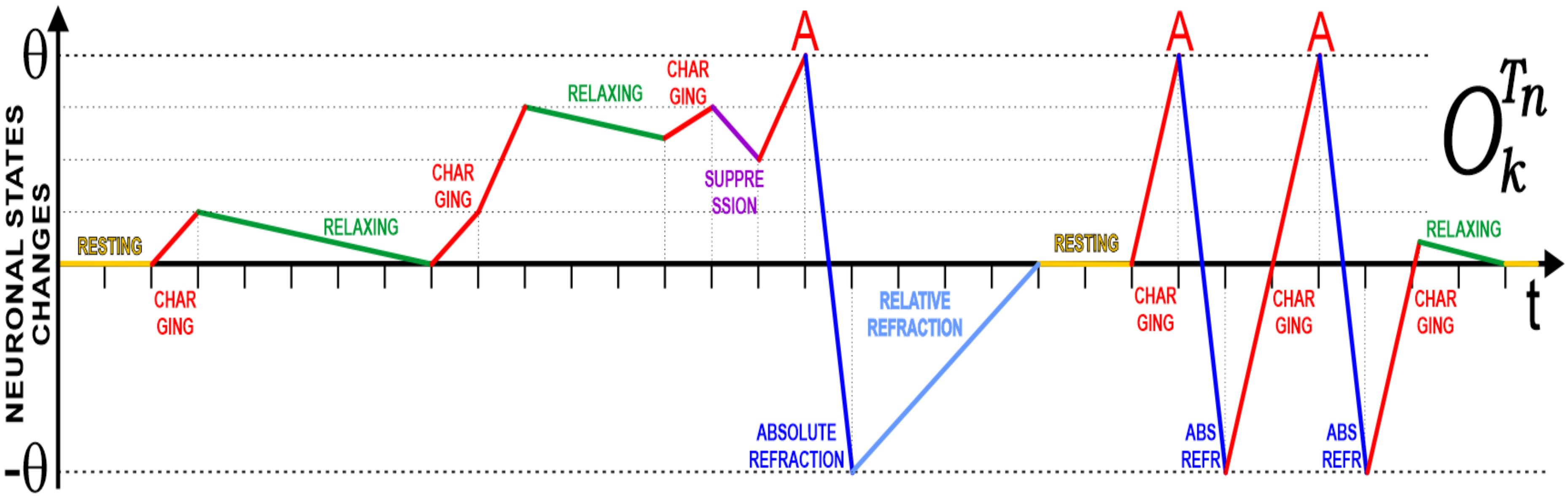
$$\theta_{O_j^{T_n}} = \begin{cases} 1 & if \quad \sum_{R_{v_i}^{a_k}} w_{R_{v_i}^{a_k}, O_j^{T_n}} \geq 1 \\ \sum_{R_{v_i}^{a_k}} w_{R_{v_i}^{a_k}, O_j^{T_n}} & if \quad \sum_{R_{v_i}^{a_k}} w_{R_{v_i}^{a_k}, O_j^{T_n}} < 1 \end{cases}$$

The above definition of the activation threshold allows for activation of an object neuron whenever it is stimulated by the **whole defining combination** of this neuron, or when it is stimulated by a **sufficiently representative subset of rare or unique features** defining this neuron, e.g. if a feature defines only one object neuron, then it is enough to recognize it when this feature appears.



Linear Approximation of the Internal Neuronal Processes

The DASNG associative spiking neurons (ASNs) uses a **linear approximation** of all processes. This greatly simplifies and speeds up calculations of neuronal states:

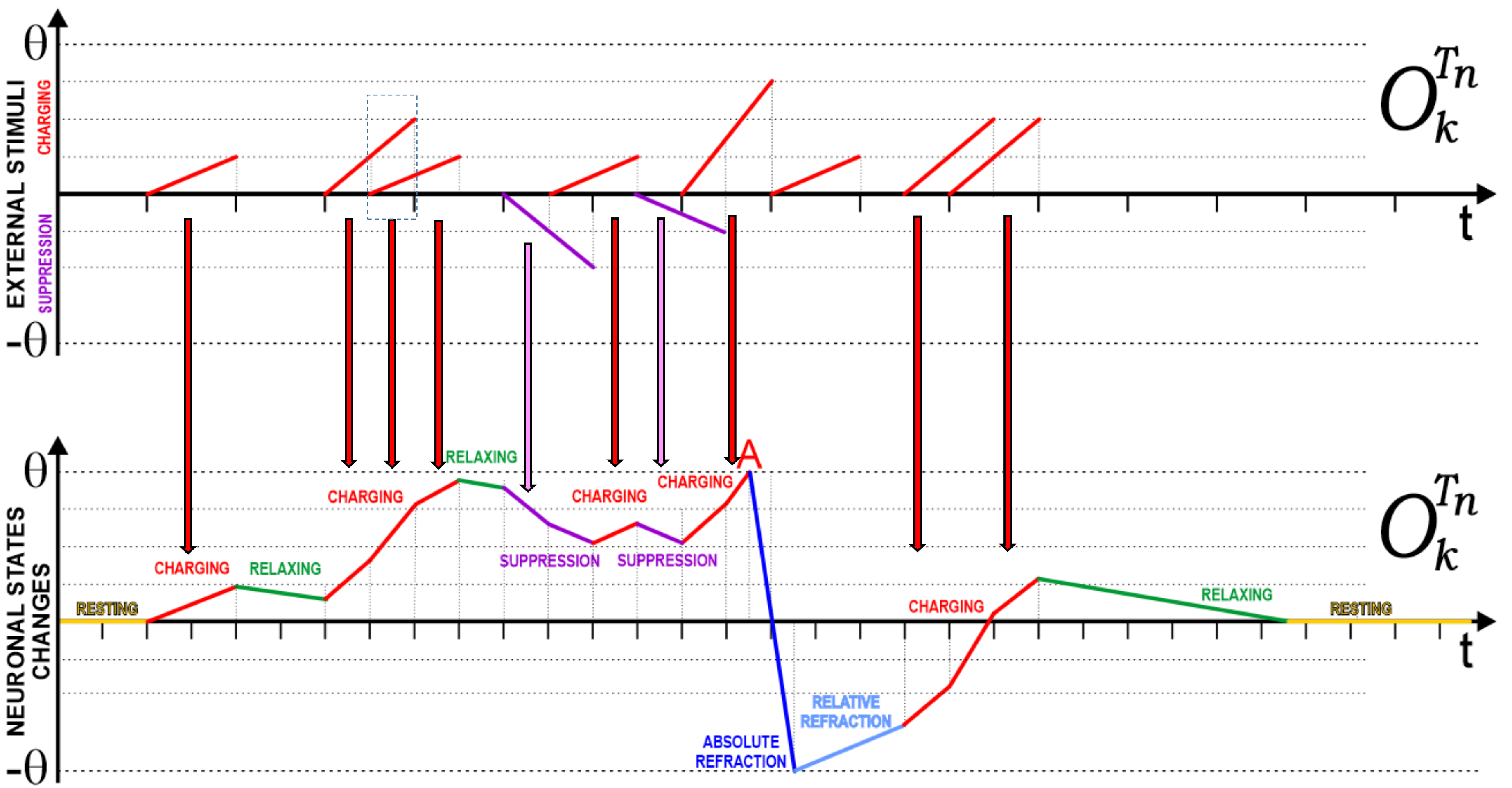


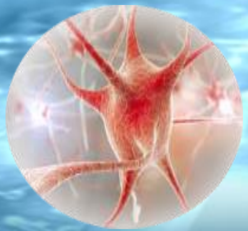
Each neuron creates an **internal neuronal process queue (IPQ)** of successive processes ordered after the time of their beginning. New processes are added to this queue on the basis of stimuli coming from other neurons or a sensor.

In order to appropriately order parallel processes of all neurons in the DASNG in time, there is used **a global event queue (GEQ)** and each event watches a single process.

Creation and Updates of the Internal Process Queue

The neuronal internal process queue (IPQ) combines external stimuli with internal processes and chronologically orders neuronal processes to not overlap in time.

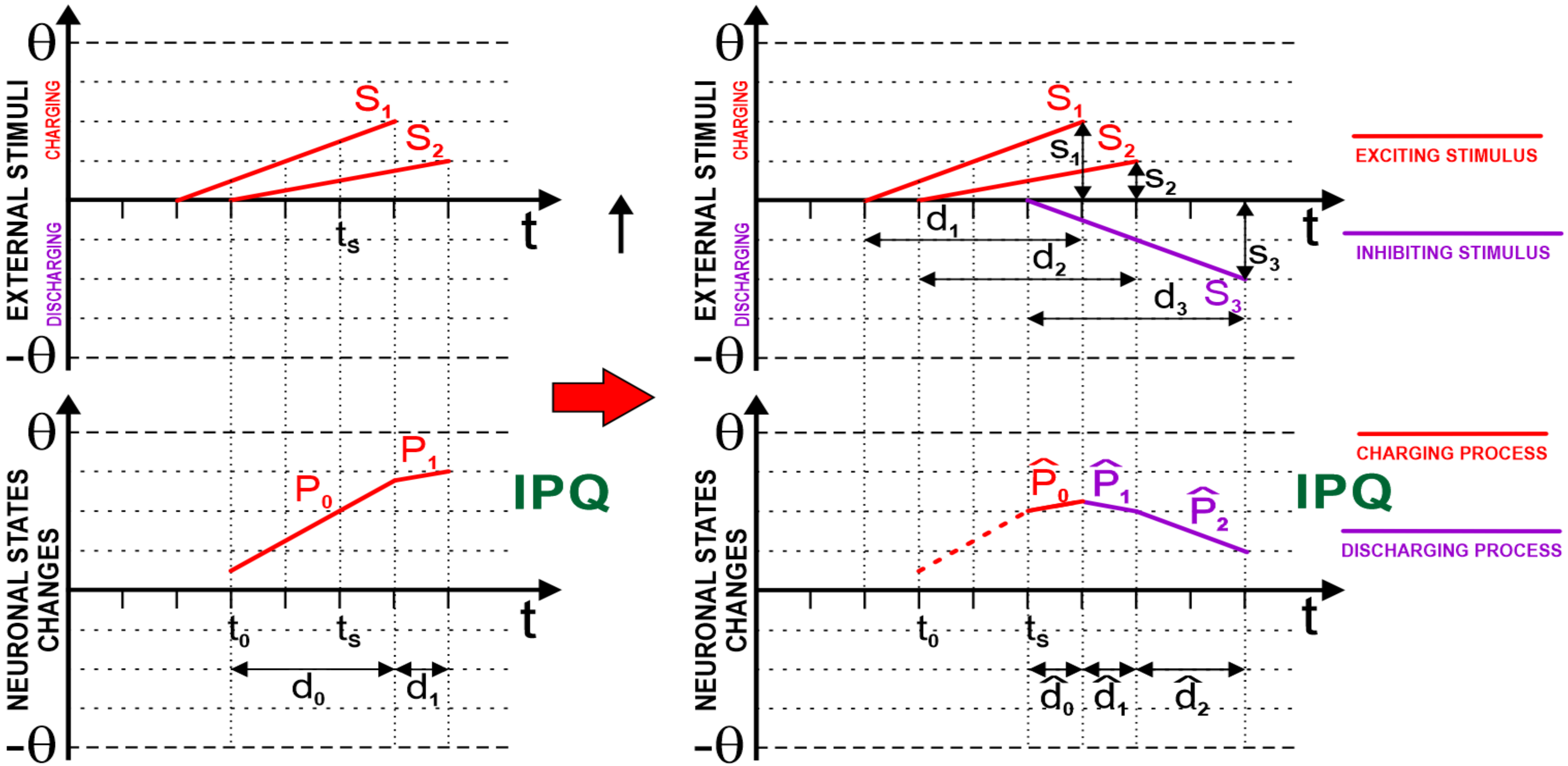


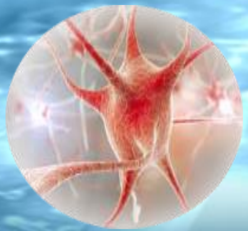


Combining the Internal Processes with a New Stimulus

$$\widehat{P}_0 = (\widehat{r}_0, t_s, d_0 - (t_s - t_0), s_0 \cdot \frac{d_0 - (t_s - t_0)}{d_0} + s_s \cdot \frac{d_0 - (t_s - t_0)}{d_s}, \widehat{p}_0)$$

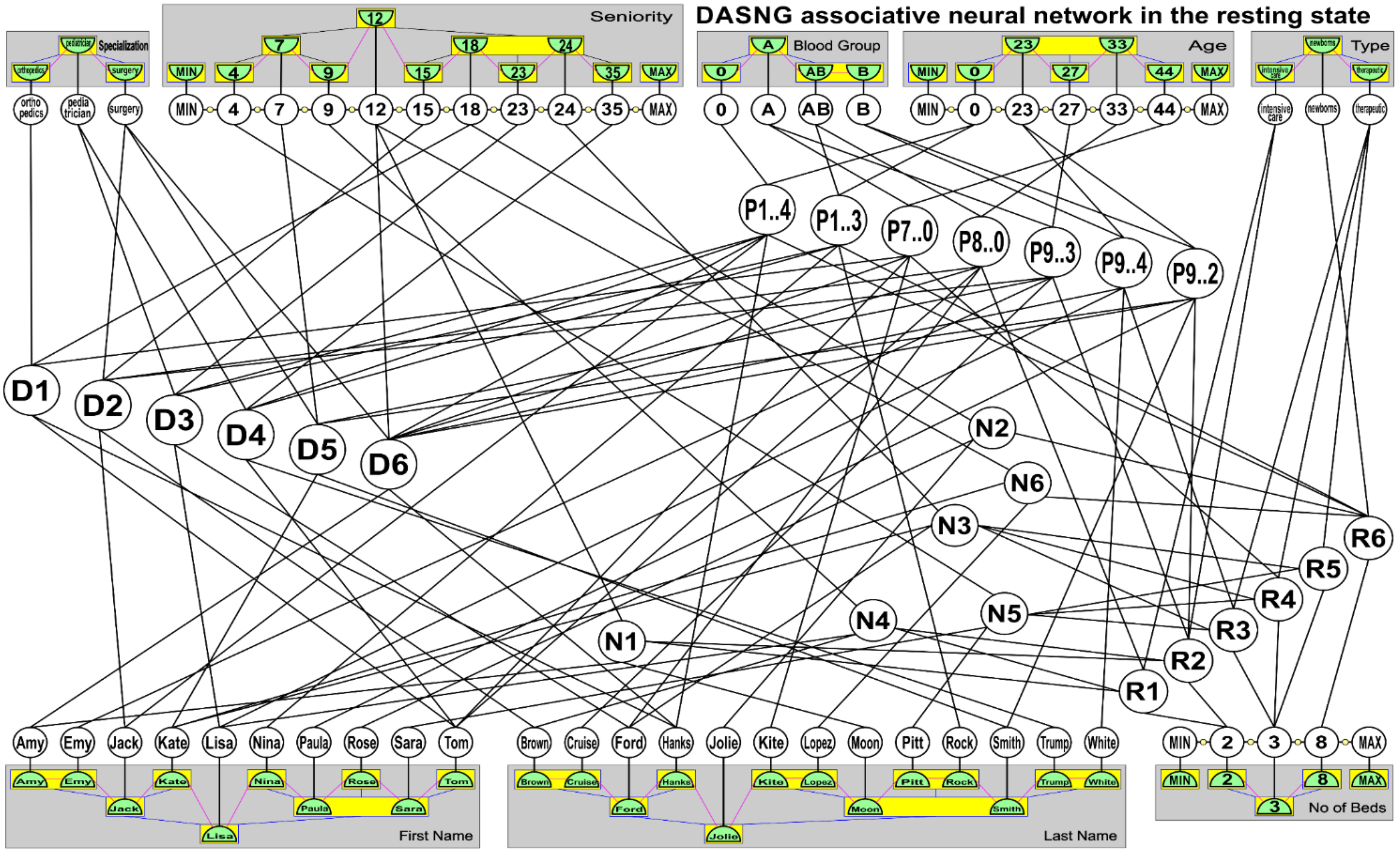
$$\widehat{P}_1 = (\widehat{r}_1, t_0 + d_0, d_s - (d_0 - (t_s - t_0)), s_s \cdot \frac{d_s - (d_0 - (t_s - t_0))}{d_s}, \widehat{p}_1)$$

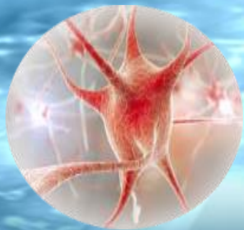




Inference using the DASNG network

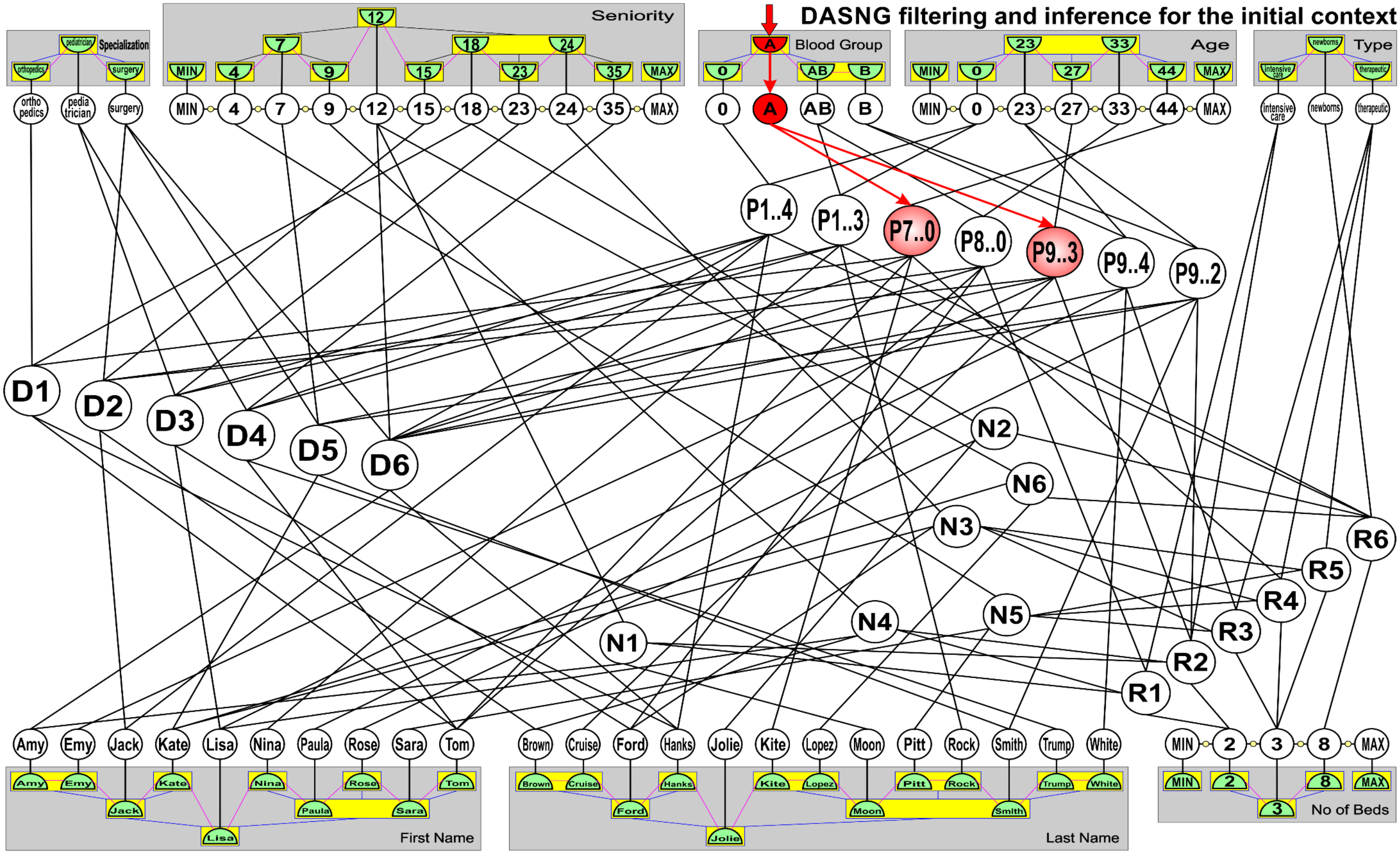
Neuronal inference can be achieved by stimulation of sensors!

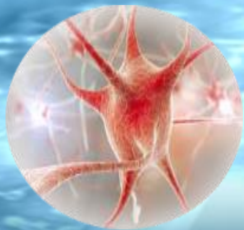




Inference using the DASNG network

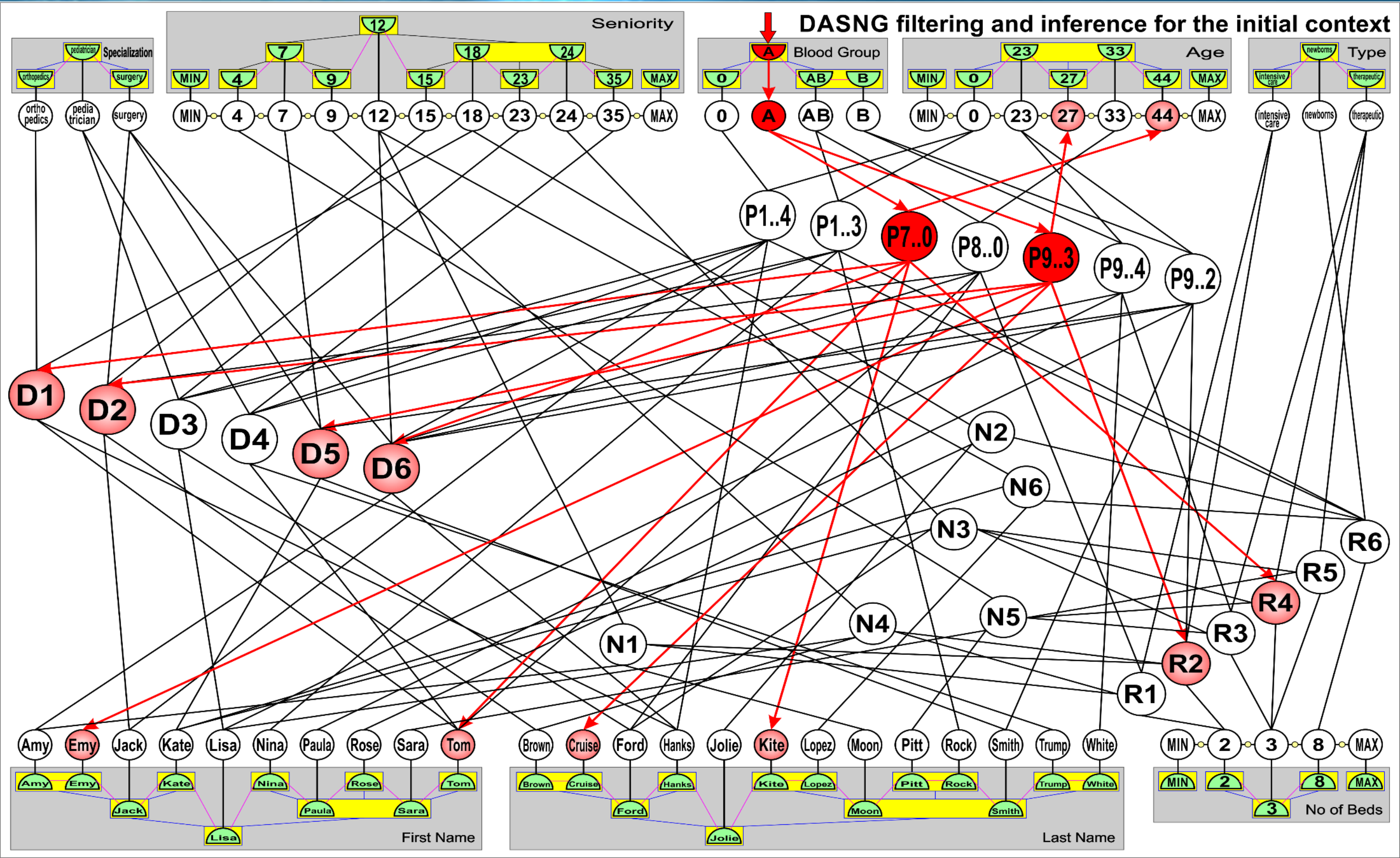
We can quickly get any associated information waiting for neuronal activity!





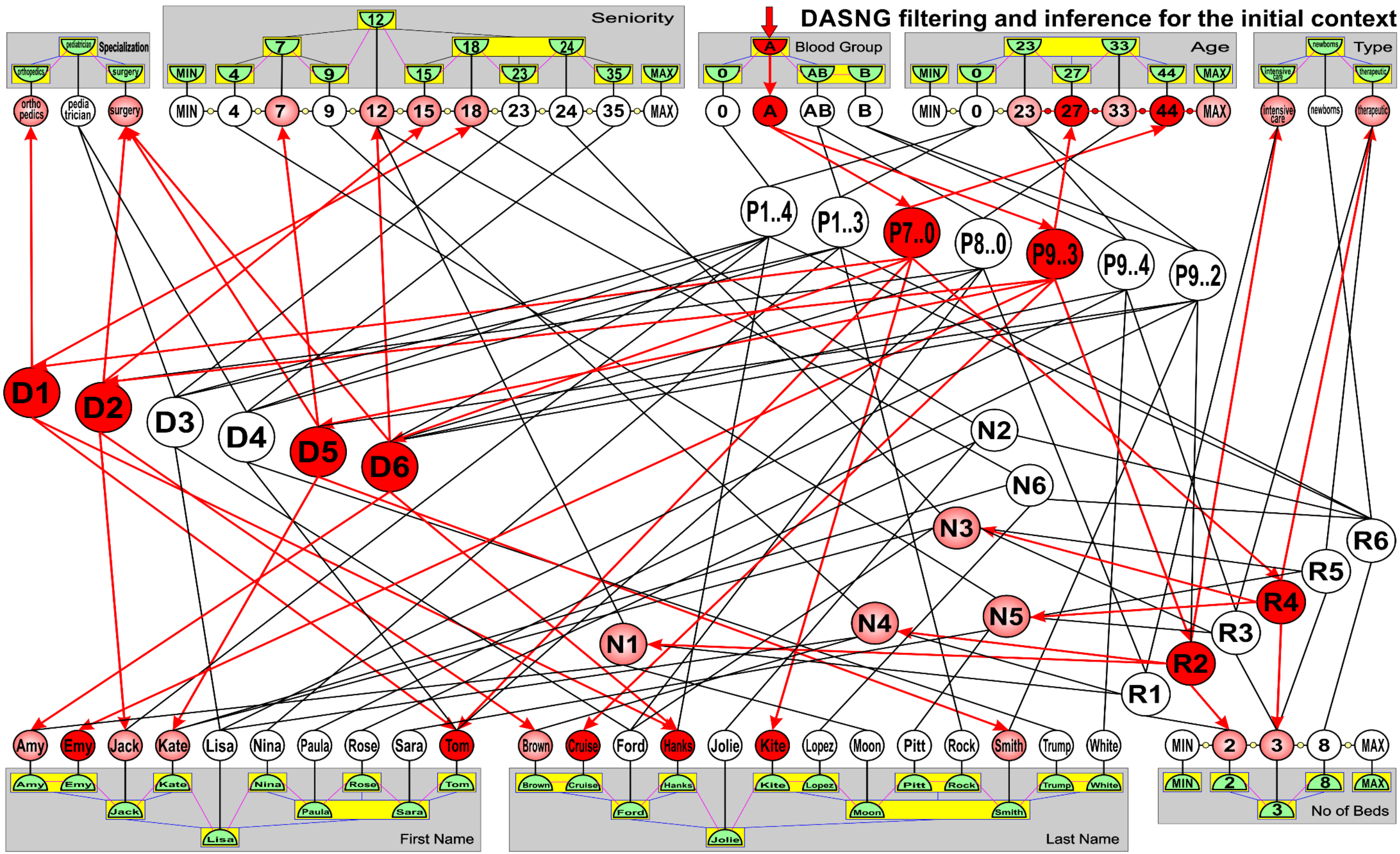
Inference using the DASNG network

Connections representing associations allow for further inferences.



Inference using the DASNG network

Indirectly associated information is also available after short time!



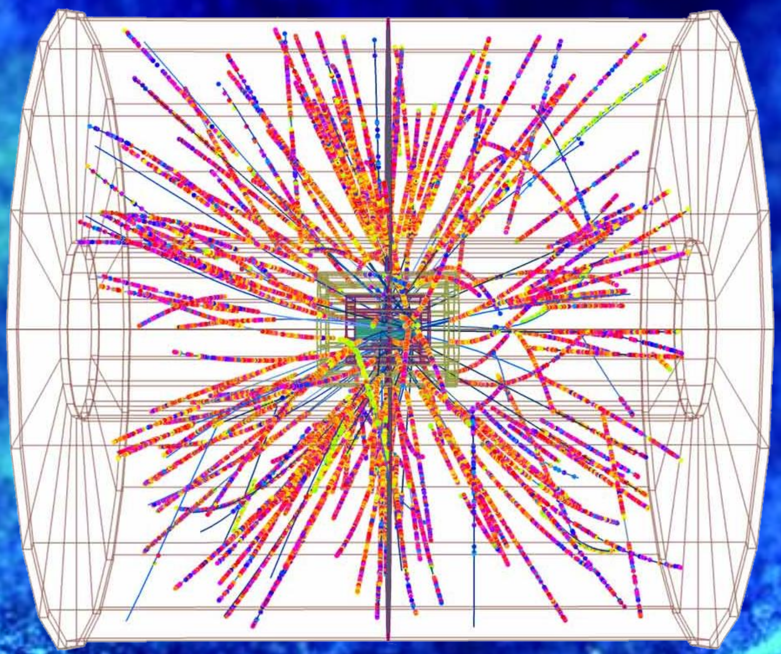
CONCLUSIONS

DEEP ASSOCIATIVE SEMANTIC NEURAL GRAPHS (DASNG) can be used to:

- **transform databases into the reactive associative data structure,**
- **create deep neural network architectures for spiking neurons,**
- **represent complex objects contextually alike in databases, additionally specifying the strength of associated (related) objects (entities),**
- **filter values or objects (entities) according to the initial stimulation(s),**
- **quickly return objects sorted after any combination of attributes,**
- **immediately get minima and maxima of any attribute,**
- **inference on the basis of the initial context used for stimulation of the DASNG network using sensors and sensory neurons,**
- **create knowledge-based cognitive and artificial intelligence systems.**

APPLICATIONS

DEEP ASSOCIATIVE SEMANTIC NEURAL GRAPHS (DASNG)
are planned to be used in CERN in
A Large Ion Collider Experiment (ALICE) in O2 and O3 run
for quality control and Big Data analysis in real-time.



DASNG can be parallelized and draw conclusions in constant time.

Questions or Remarks?

1. **A. Horzyk**, J. A. Starzyk, J. Graham, Integration of Semantic and Episodic Memories, IEEE Transactions on Neural Networks and Learning Systems, 2017, DOI: 10.1109/TNNLS.2017.2728203.
2. **A. Horzyk** and J.A. Starzyk, Fast Neural Network Adaptation with Associative Pulsing Neurons, IEEE Xplore, In: 2017 IEEE Symposium Series on Computational Intelligence, 2017.
3. Basawaraj, Janusz A. Starzyk and **A. Horzyk**, Lumped Mini-Column Associative Knowledge Graphs, IEEE Xplore, In: 2017 IEEE Symposium Series on Computational Intelligence, 2017.
4. **A. Horzyk**, Deep Associative Semantic Neural Graphs for Knowledge Representation and Fast Data Exploration, Proc. of KEOD 2017, SCITEPRESS Digital Library, 2017.
5. **A. Horzyk**, Neurons Can Sort Data Efficiently, Proc. of ICAISC 2017, Springer-Verlag, LNAI, 2017, pp. 64-74, ICAISC BEST PAPER AWARD 2017 sponsored by Springer.
6. **A. Horzyk**, J. A. Starzyk and Basawaraj, Emergent creativity in declarative memories, IEEE Xplore, In: 2016 IEEE Symposium Series on Computational Intelligence, Greece, Athens: Institute of Electrical and Electronics Engineers, Curran Associates, Inc. 57 Morehouse Lane Red Hook, NY 12571 USA, 2016, ISBN 978-1-5090-4239-5, pp. 1-8, DOI: 10.1109/SSCI.2016.7850029.
7. **A. Horzyk**, Human-Like Knowledge Engineering, Generalization and Creativity in Artificial Neural Associative Systems, Springer-Verlag, AISC 11156, ISSN 2194-5357, ISBN 978-3-319-19089-1, ISBN 978-3-319-19090-7 (eBook), DOI 10.1007/978-3-319-19090-7, Springer, Switzerland, 2016, pp. 39-51.
8. **A. Horzyk**, Innovative Types and Abilities of Neural Networks Based on Associative Mechanisms and a New Associative Model of Neurons - Invited talk at ICAISC 2015, Springer-Verlag, LNAI 9119, 2015, pp. 26-38, DOI 10.1007/978-3-319-19324-3_3.
9. **Horzyk, A.**, How Does Generalization and Creativity Come into Being in Neural Associative Systems and How Does It Form Human-Like Knowledge?, Neurocomputing, 2014.
10. **Horzyk, A.**, Human-Like Knowledge Engineering, Generalization and Creativity in Artificial Neural Associative Systems, Springer, AISC 11156, 2014.

Google: Horzyk

AGH

AGH University of Science and Technology in Krakow

KEOD 2017
9th International Conference on Knowledge Engineering and Ontology Development
Funchal, Madeira - Portugal
1 - 3 November, 2017