Workshop on Grand Challenges of Computational Intelligence (Cyprus, September 14, 2012)

Scalability Improvement of Genetics-Based Machine Learning to Large Data Sets

Hisao Ishibuchi

Osaka Prefecture University, Japan

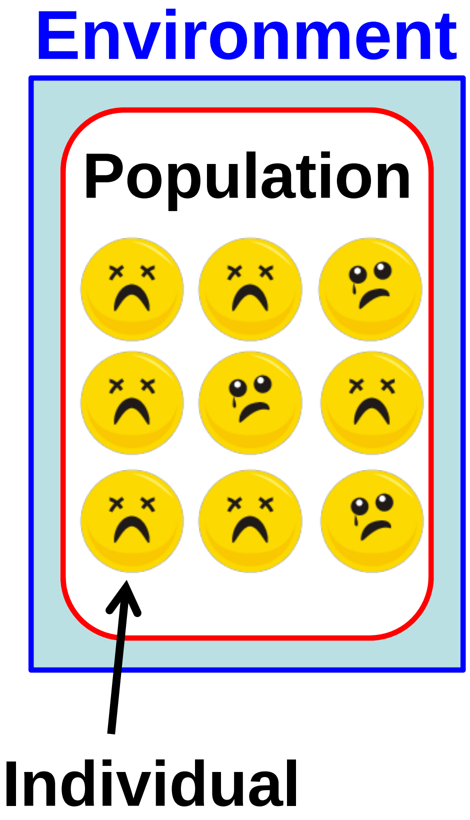
Contents of This Presentation

1. Basic Idea of Evolutionary Computation
2. Genetics-Based Machine Learning
3. Parallel Distributed Implementation
4. Computation Experiments
5. Conclusion

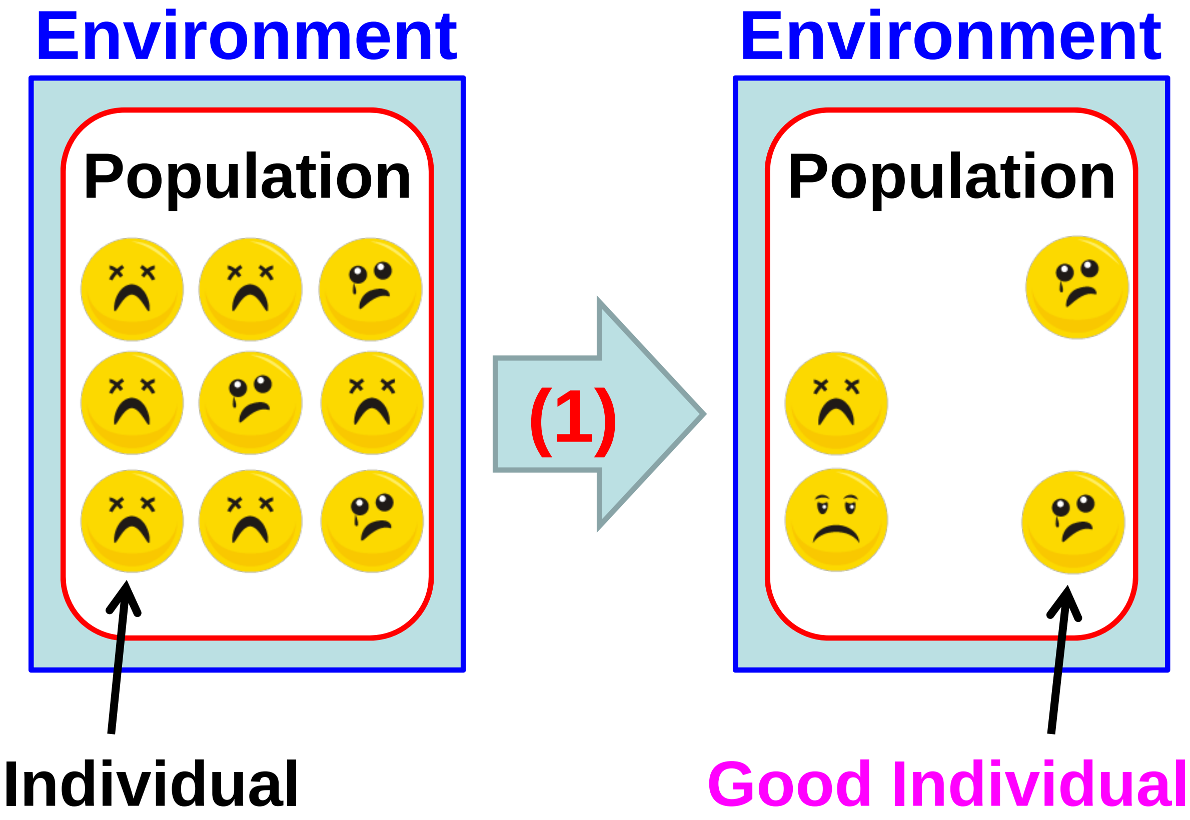
Contents of This Presentation

1. Basic Idea of Evolutionary Computation
2. Genetics-Based Machine Learning
3. Parallel Distributed Implementation
4. Computation Experiments
5. Conclusion

Basic Idea of Evolutionary Computation

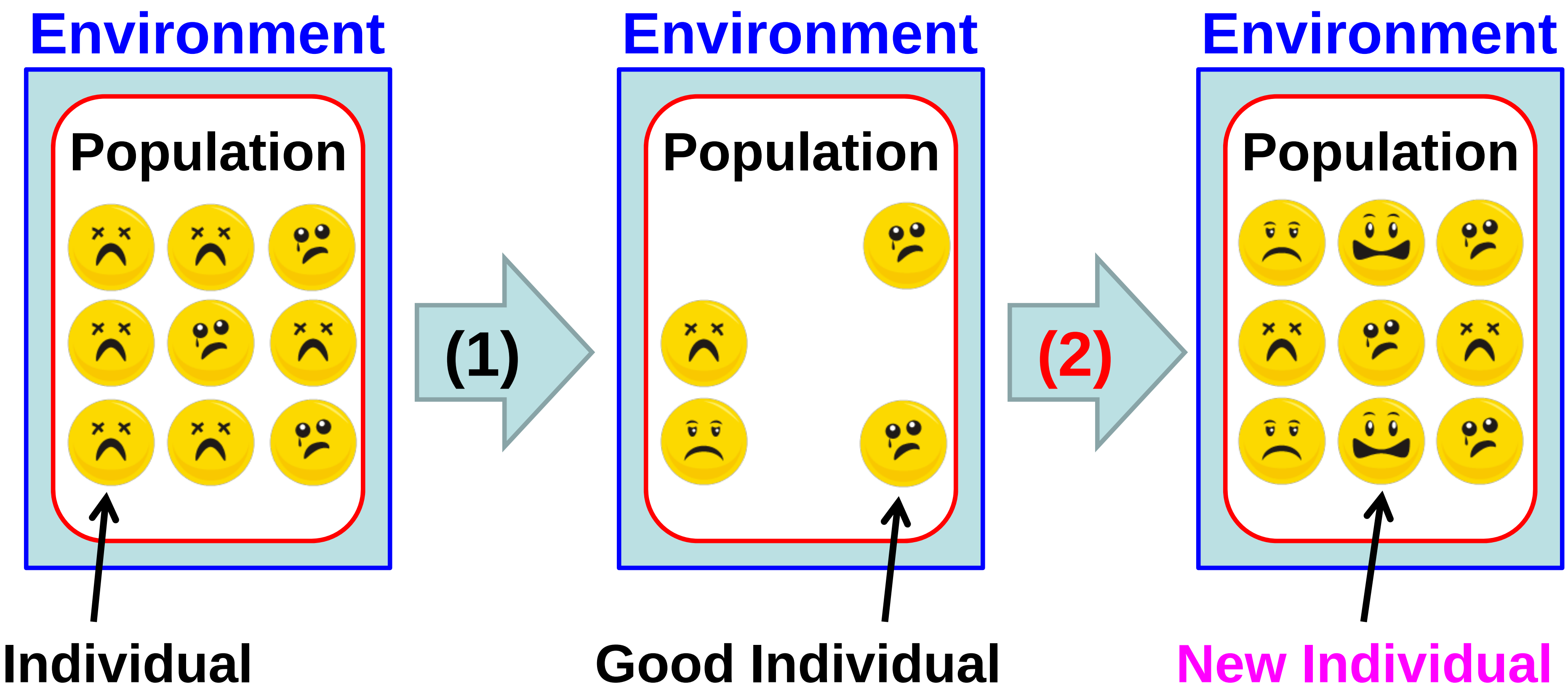


Basic Idea of Evolutionary Computation



(1) Natural selection in a tough environment.

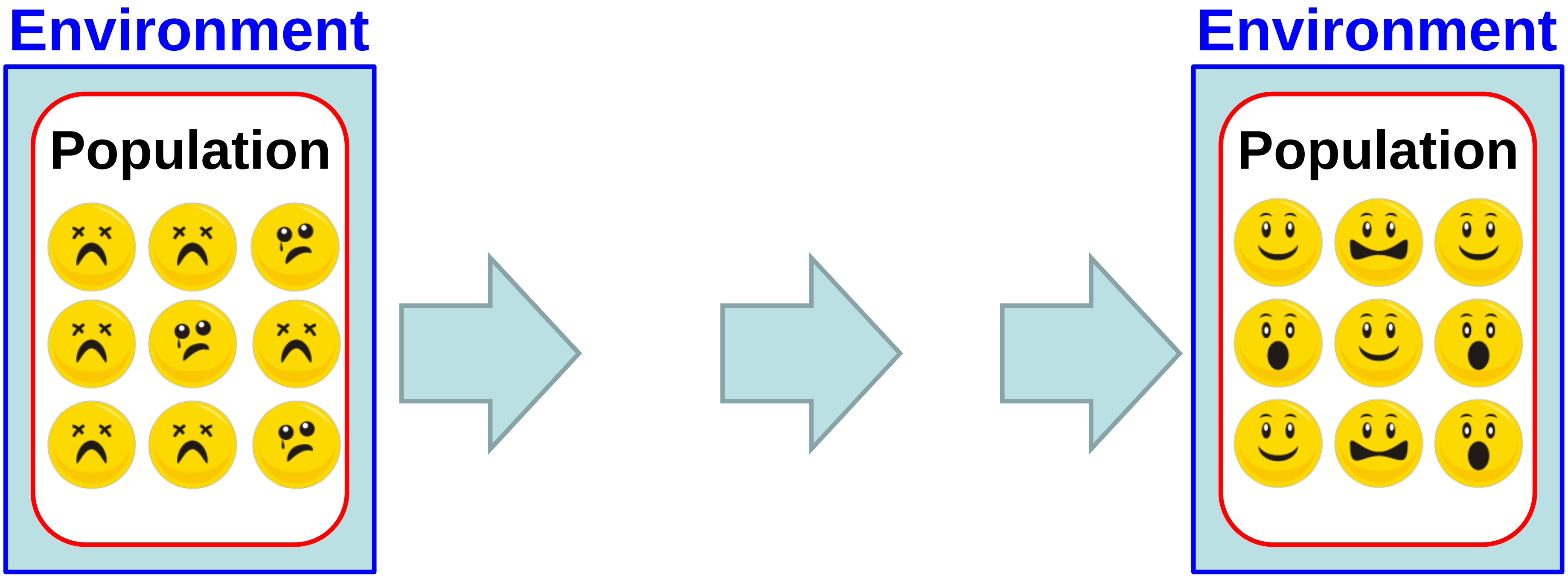
Basic Idea of Evolutionary Computation



(1) Natural selection in a tough environment.

(2) Reproduction of new individuals by crossover and mutation.

Basic Idea of Evolutionary Computation

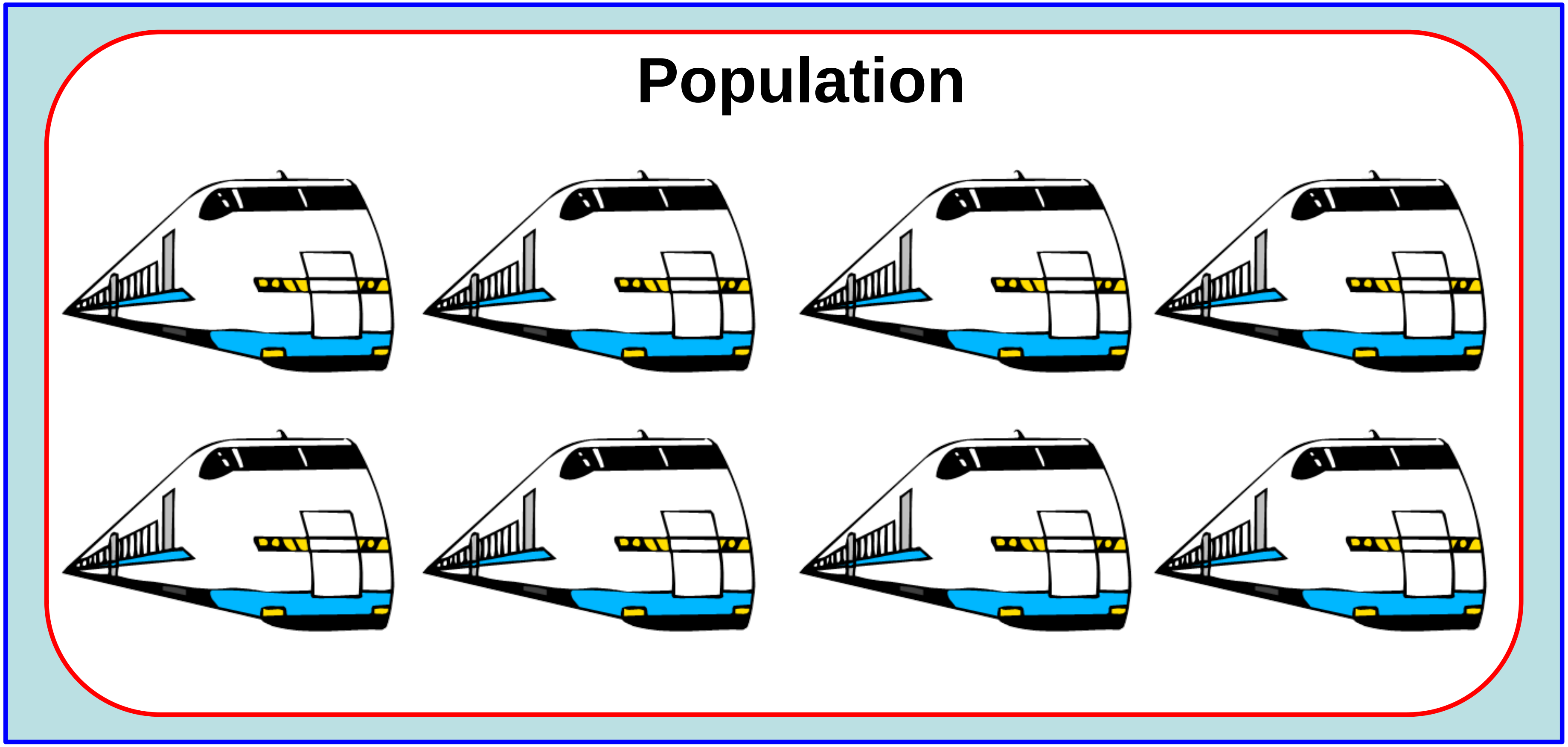


Iteration of the generation update many times

(1) Natural selection in a tough environment.
(2) Reproduction of new individuals by crossover and mutation.

Applications of Evolutionary Computation
Design of High Speed Trains

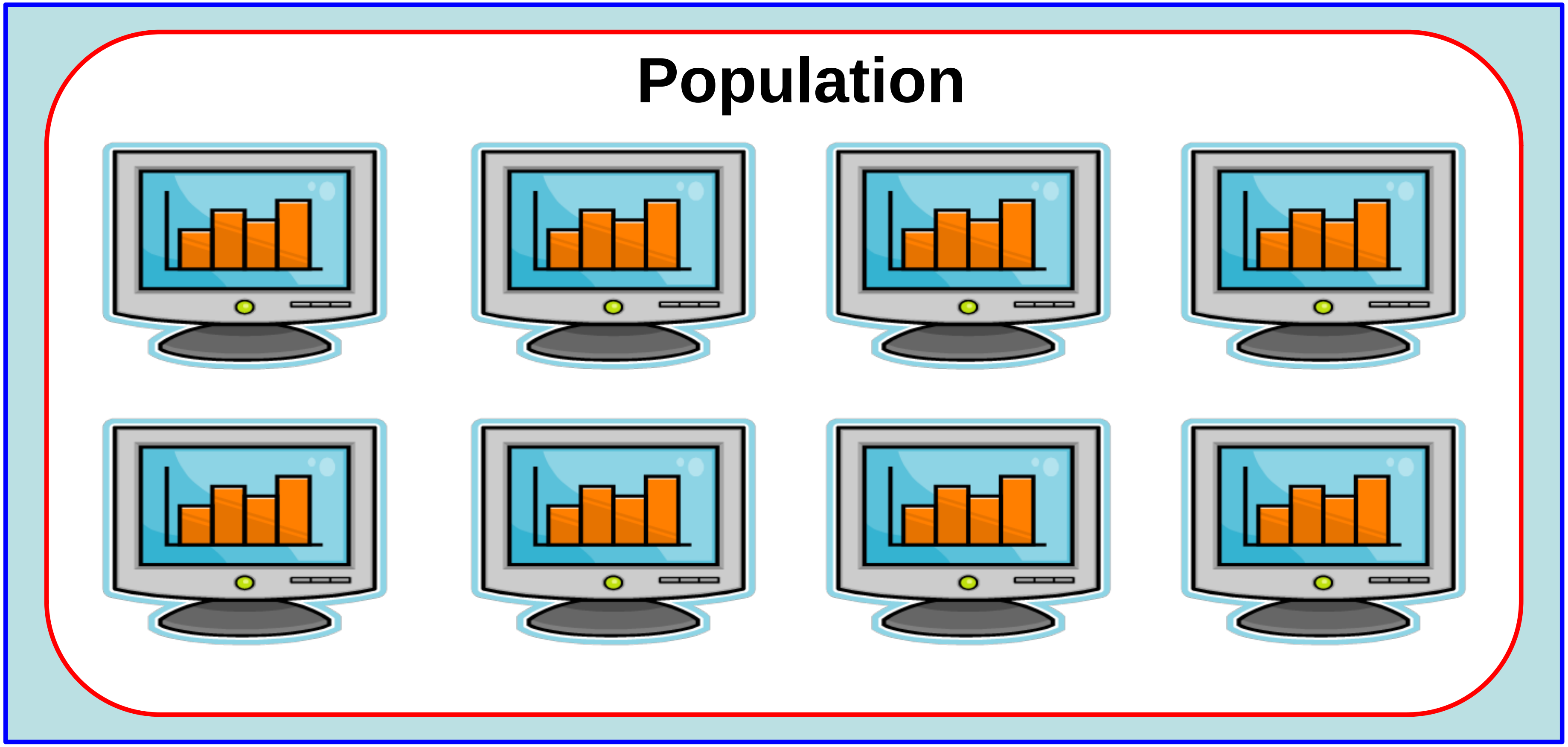
Environment



Individual = Design ( **)**

Applications of Evolutionary Computation
Design of Stock Trading Algorithms

Environment

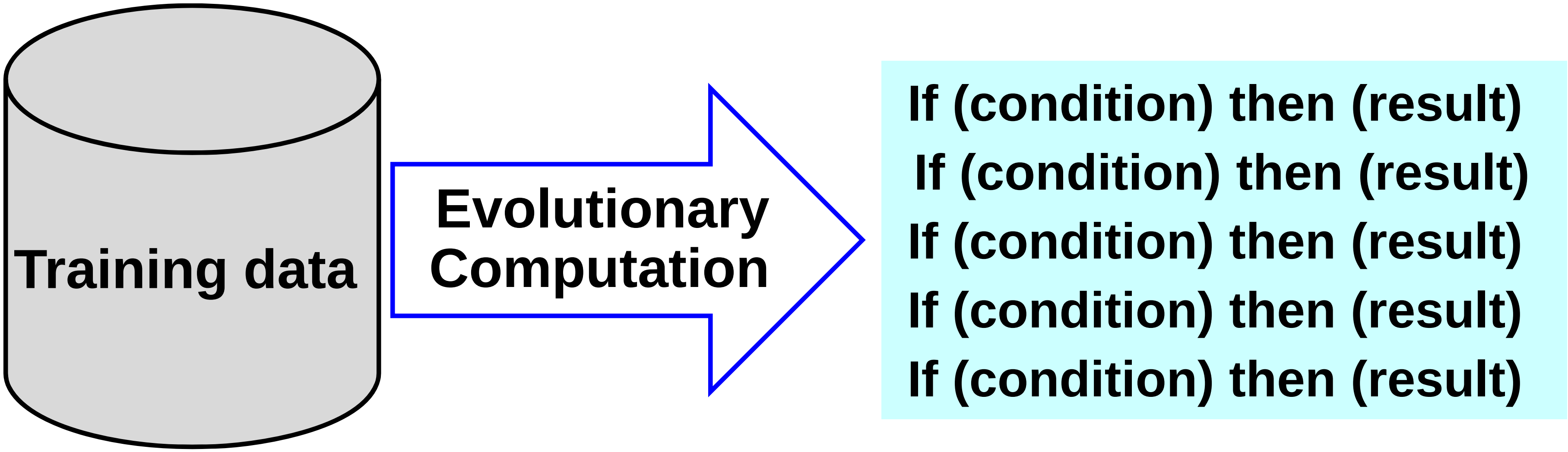


Individual = Trading Algorithm ( **)**

Contents of This Presentation

1. Basic Idea of Evolutionary Computation
2. Genetics-Based Machine Learning
3. Parallel Distributed Implementation
4. Computation Experiments
5. Conclusion

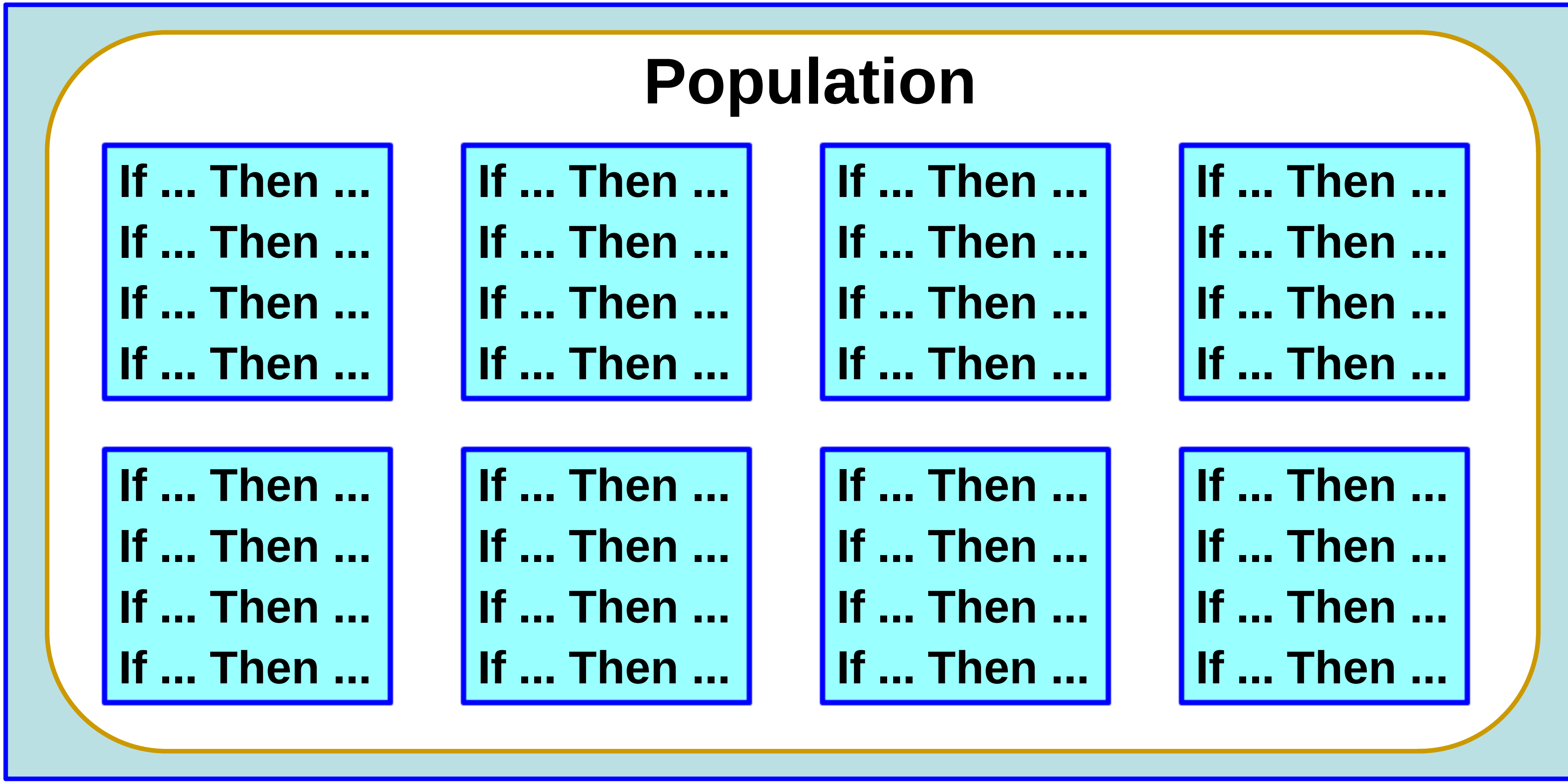
Genetics-Based Machine Learning
Knowledge Extraction from Numerical Data



Design of Rule-Based Systems

Design of Rule-Based Systems

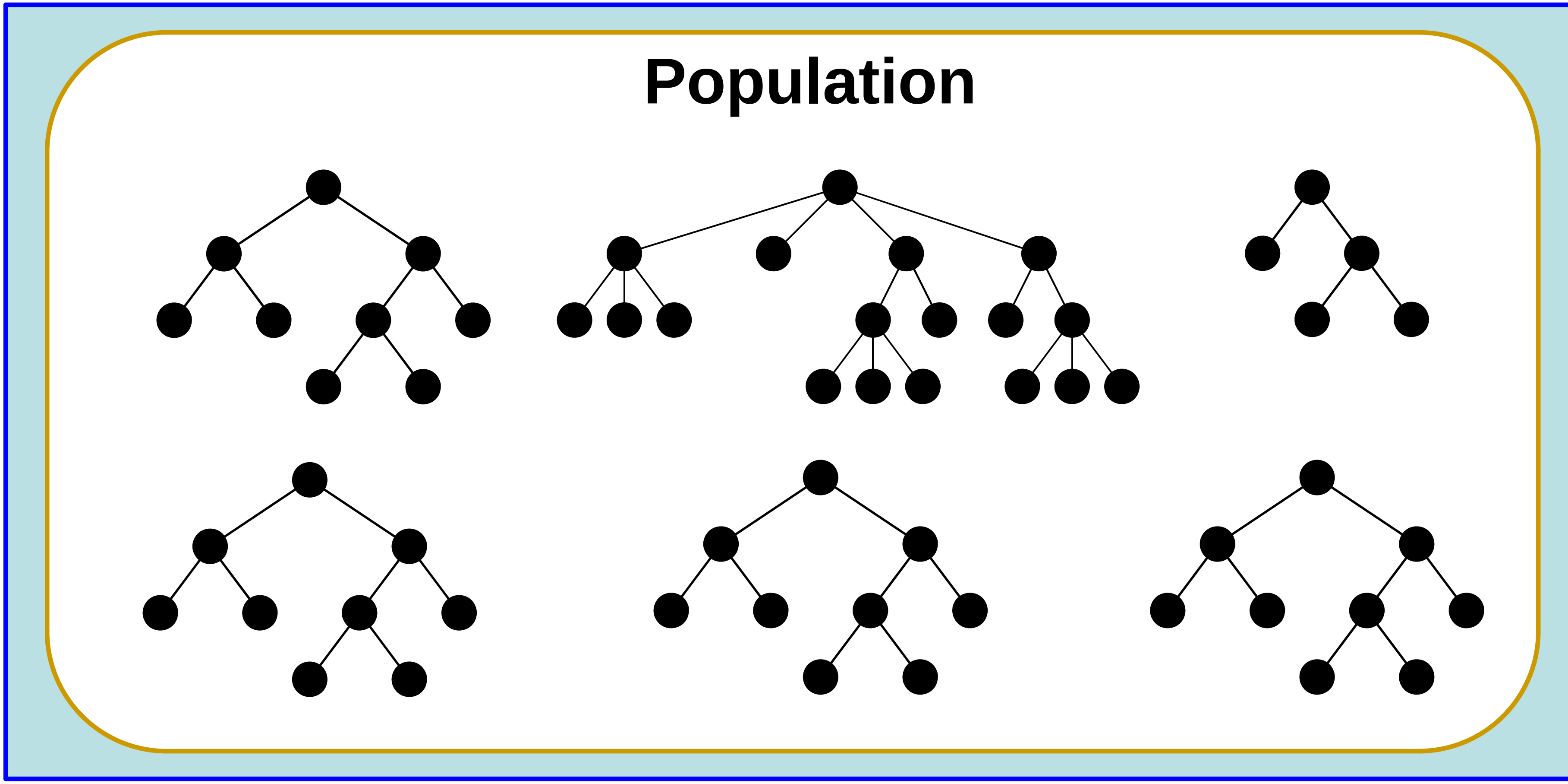
Environment

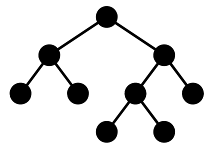


Individual = Rule-Based System (If ... Then ... If ... Then ... If ... Then ... If ... Then ... **)**

Design of Decision Trees

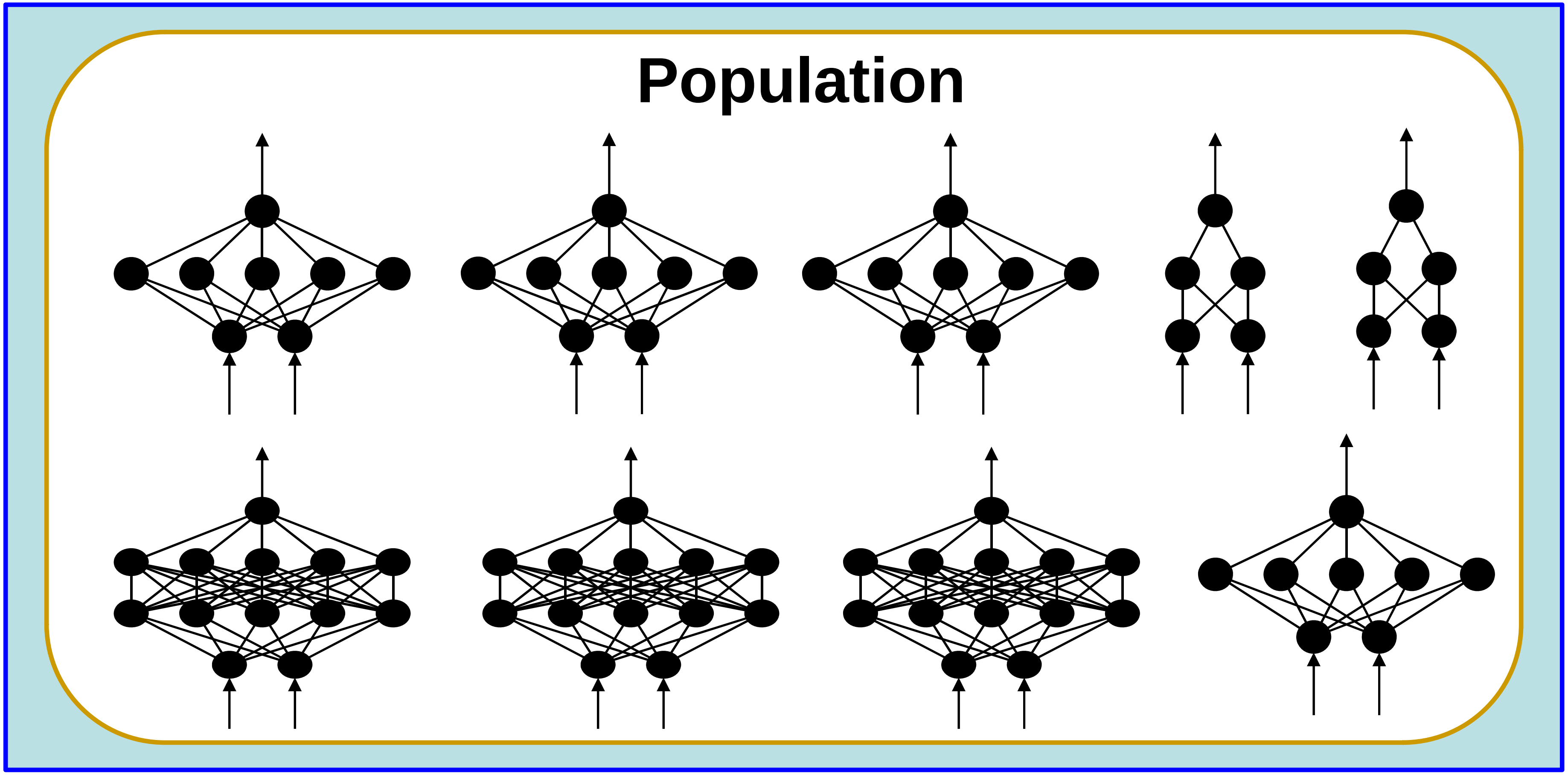
Environment

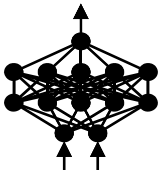


Individual = Decision Tree ( **)**

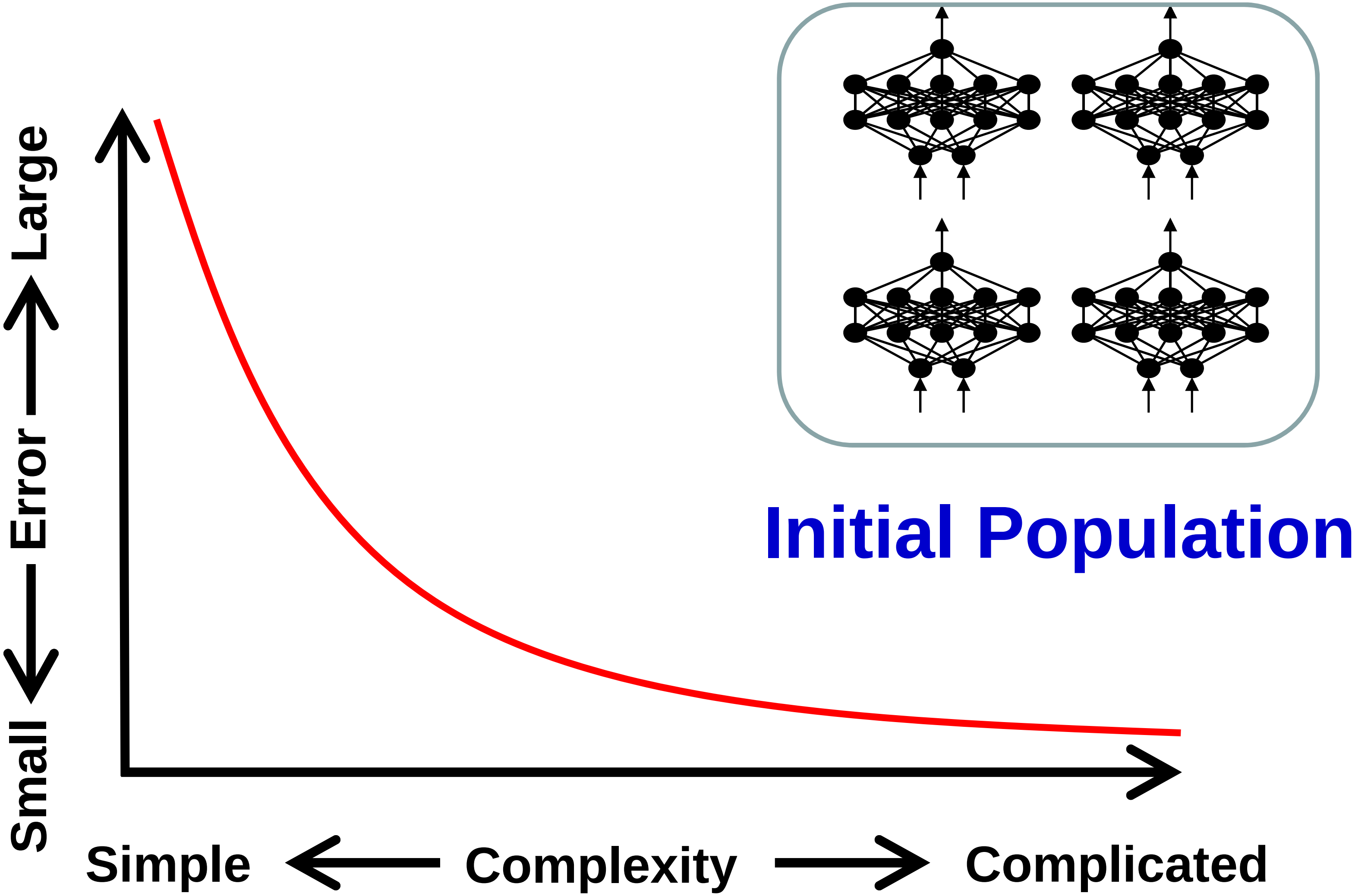
Design of Neural Networks

Environment



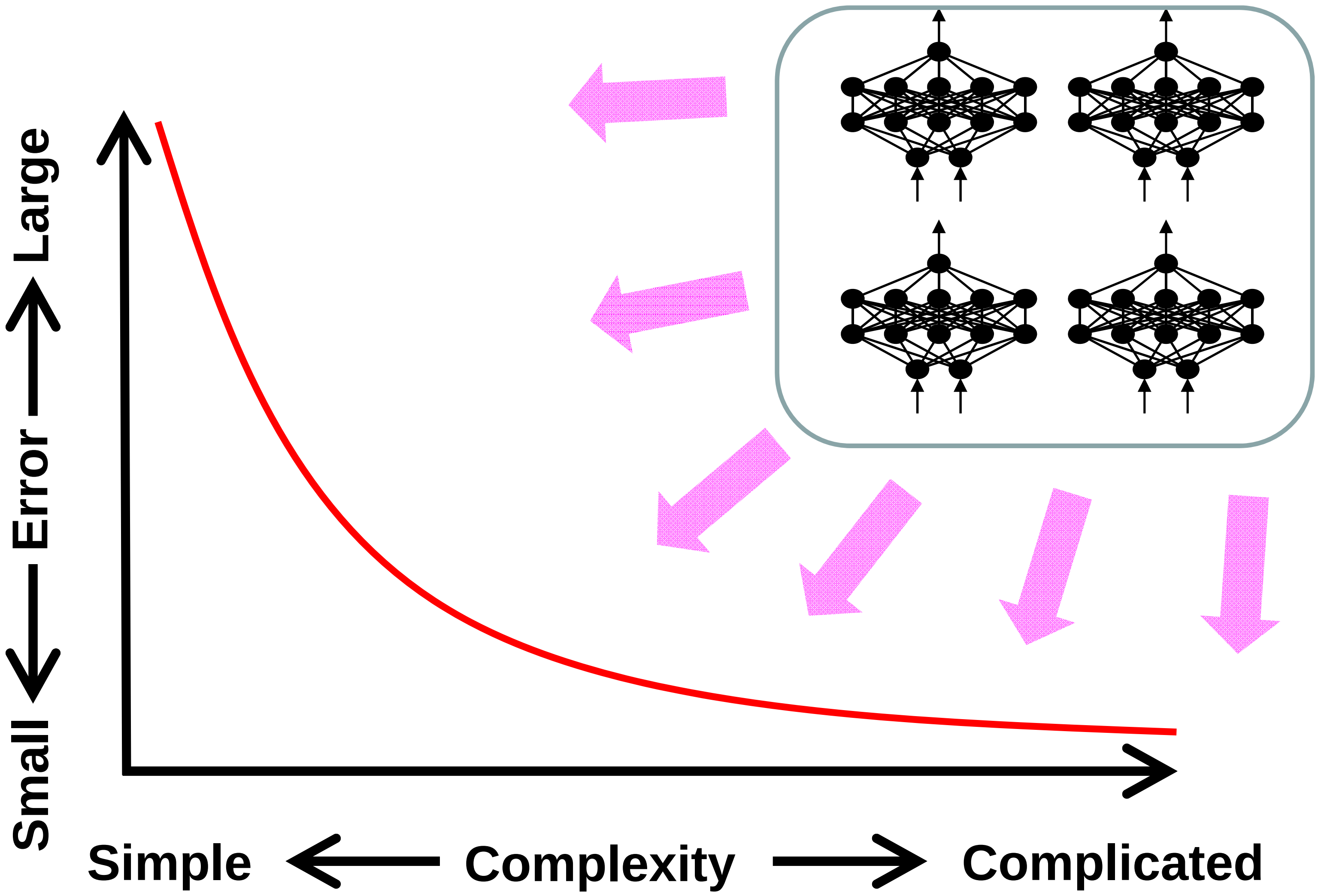
Individual = Neural Network ( **)**

Multi-Objective Evolution
Minimization of Errors and Complexity



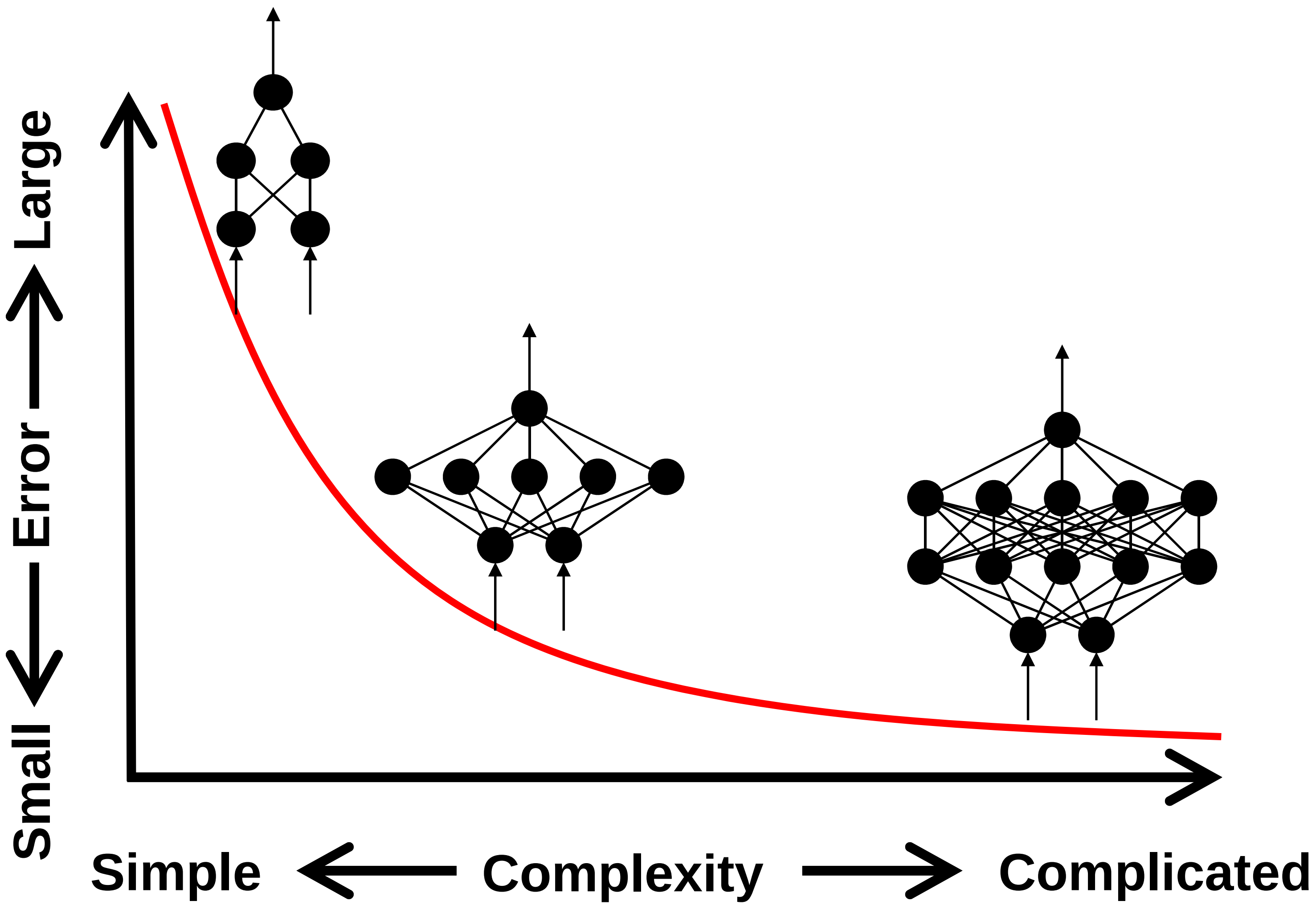
Multi-Objective Evolution

Minimization of Errors and Complexity

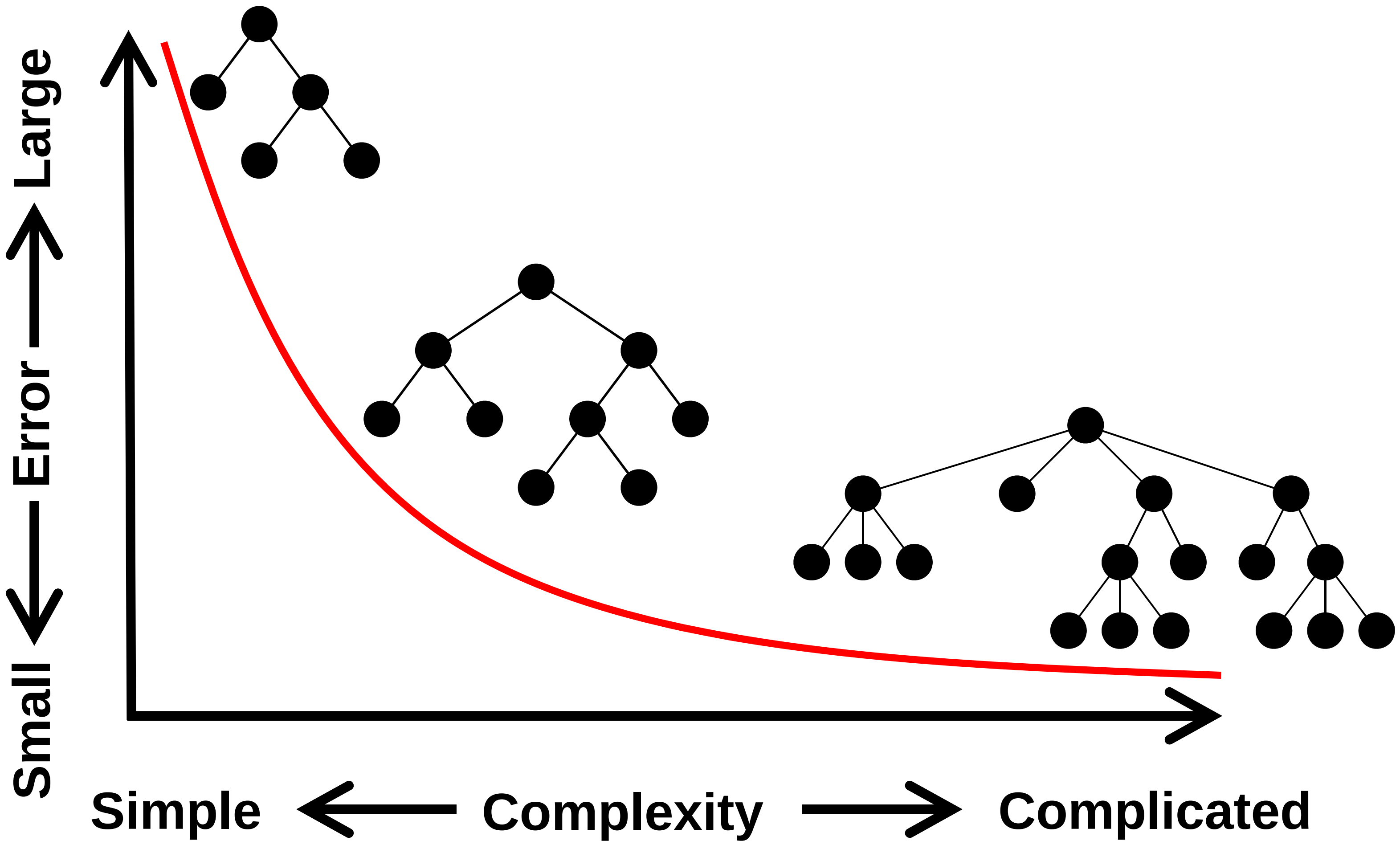


Multi-Objective Evolution

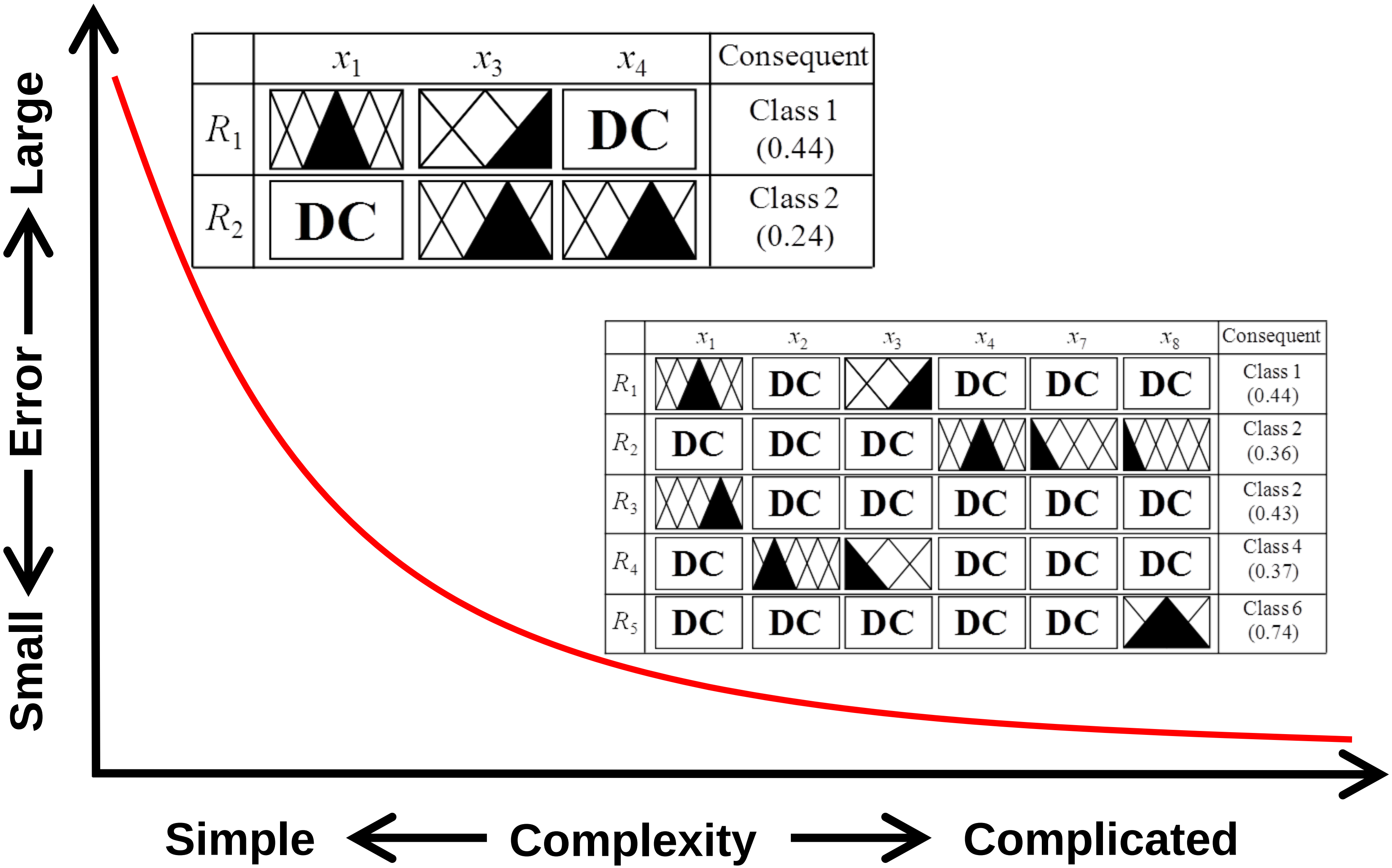
A number of different neural networks



Multi-Objective Evolution
A number of different decision trees



Multi-Objective Evolution
A number of fuzzy rule-based systems



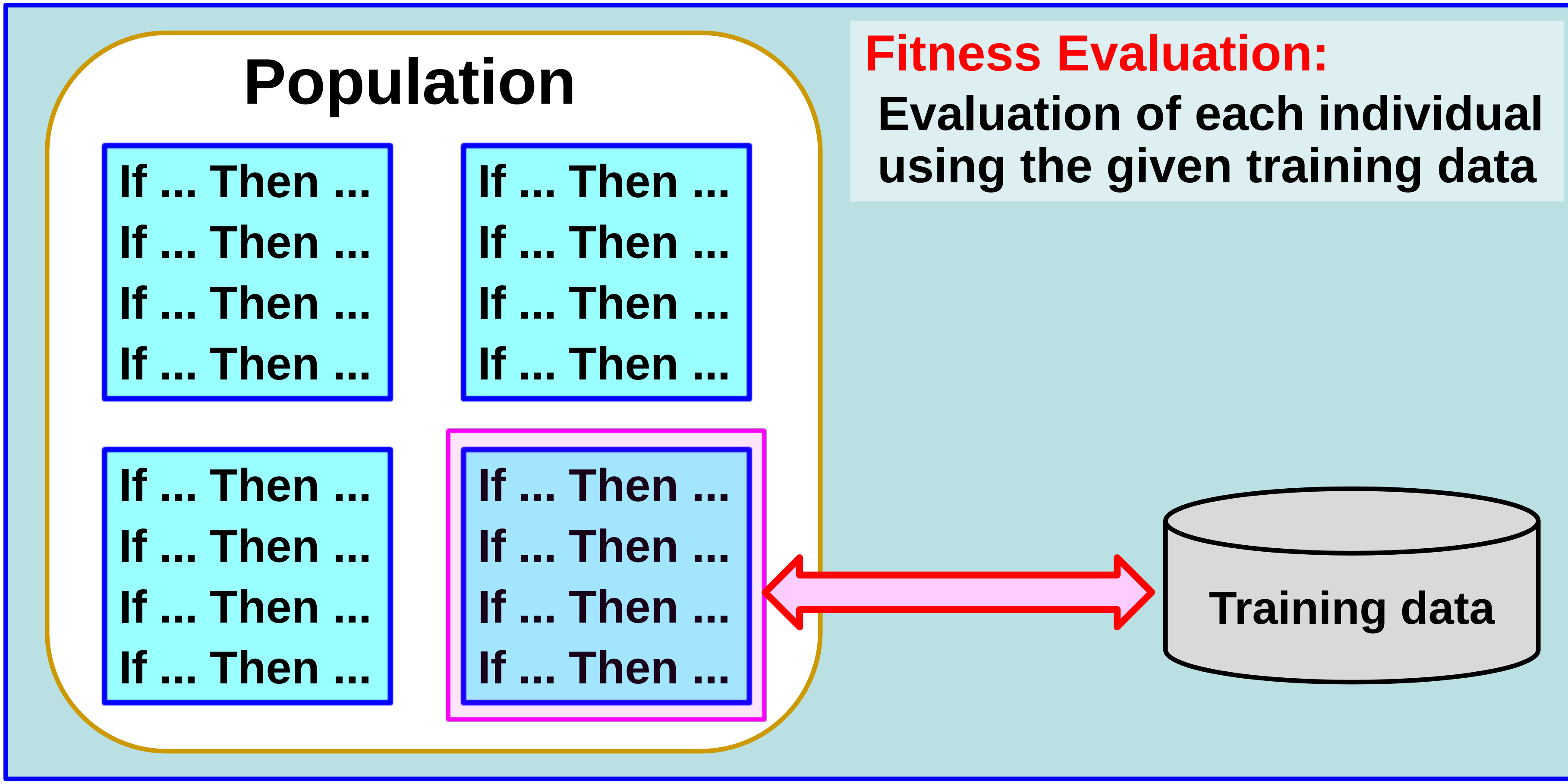
Contents of This Presentation

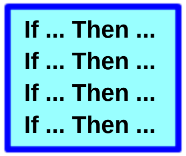
1. Basic Idea of Evolutionary Computation
2. Genetics-Based Machine Learning
3. Parallel Distributed Implementation
4. Computation Experiments
5. Conclusion

Difficulty in Applications to Large Data

Computation Load for Fitness Evaluation

Environment



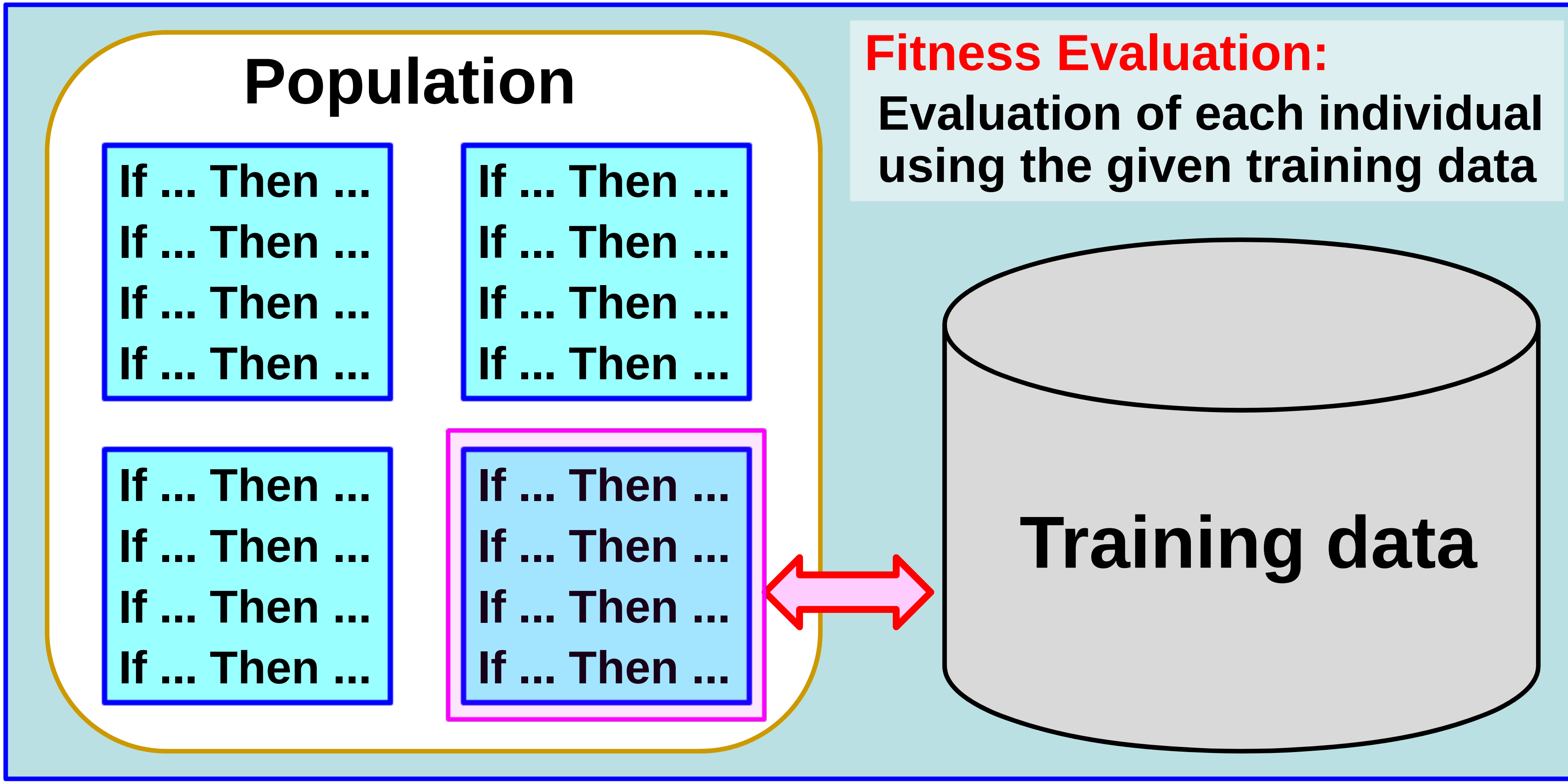
Individual = Rule-Based System (

)

Difficulty in Applications to Large Data

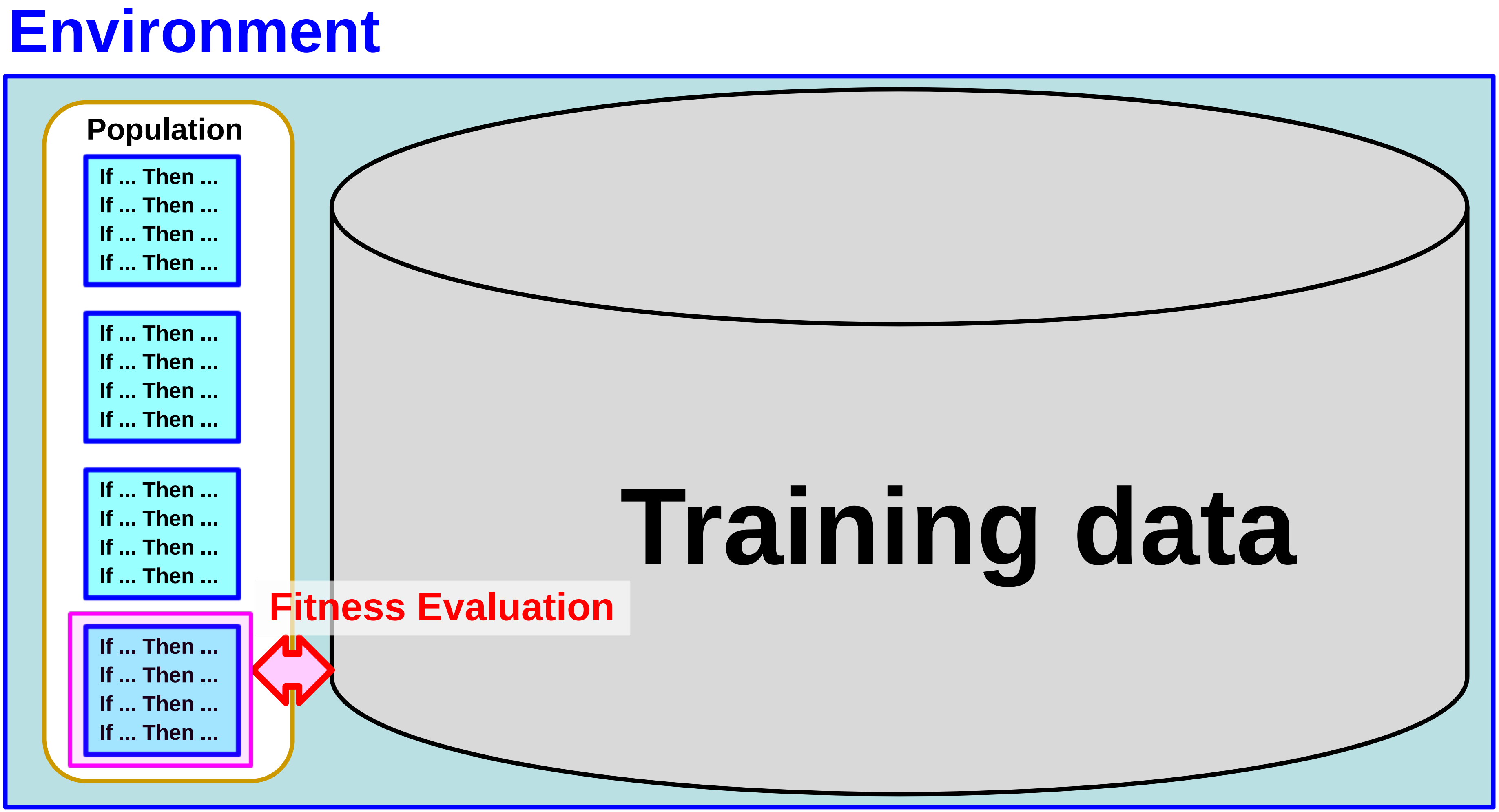
Computation Load for Fitness Evaluation

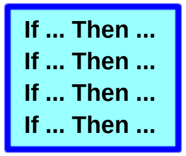
Environment



Individual = Rule-Based System (If ... Then ... If ... Then ... If ... Then ... If ... Then ... **)**

Difficulty in Applications to Large Data
Computation Load for Fitness Evaluation

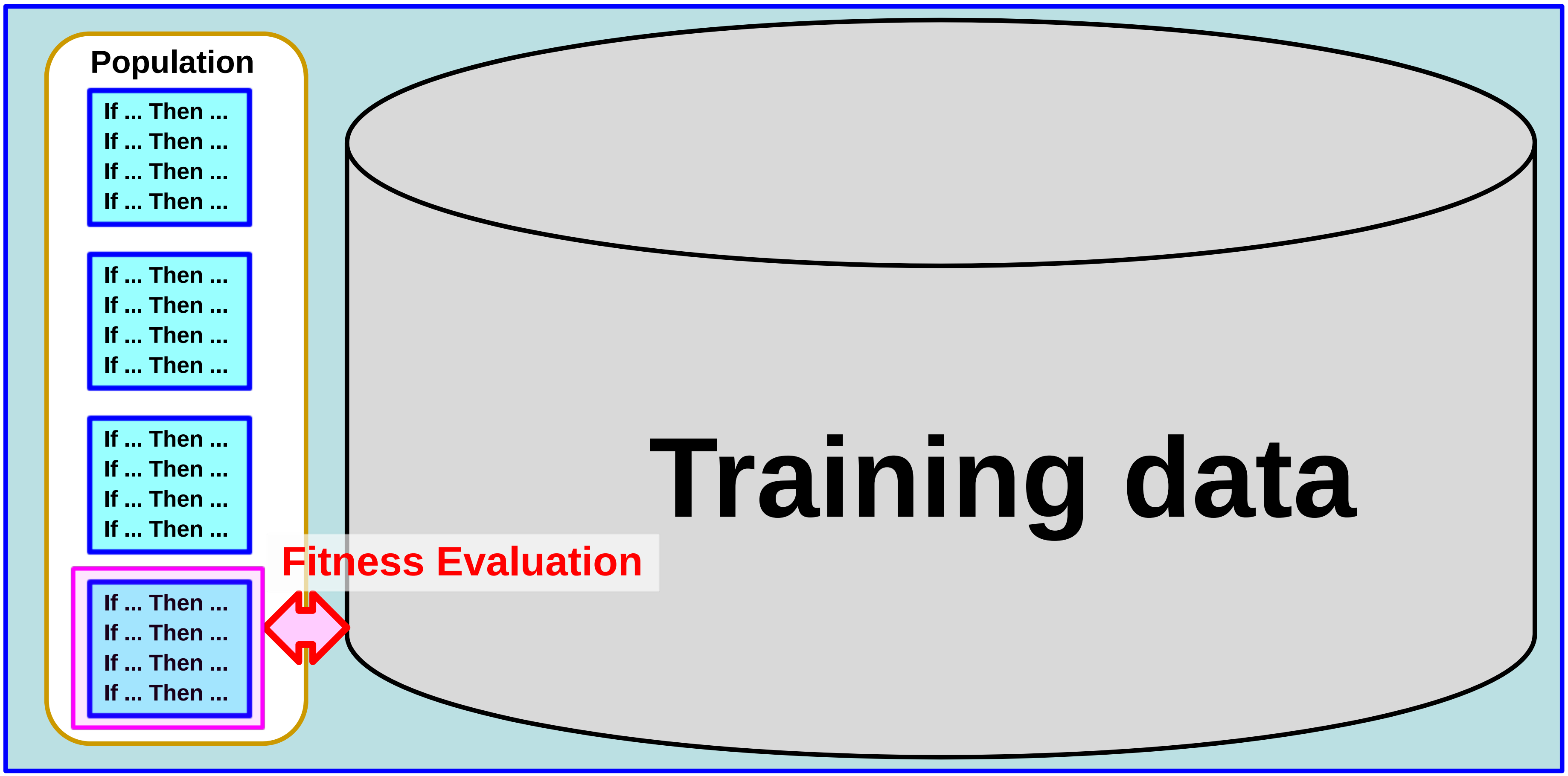


Individual = Rule-Based System ( **)**

The Main Issue in This Presentation

How to Decrease the Computation Load

Environment

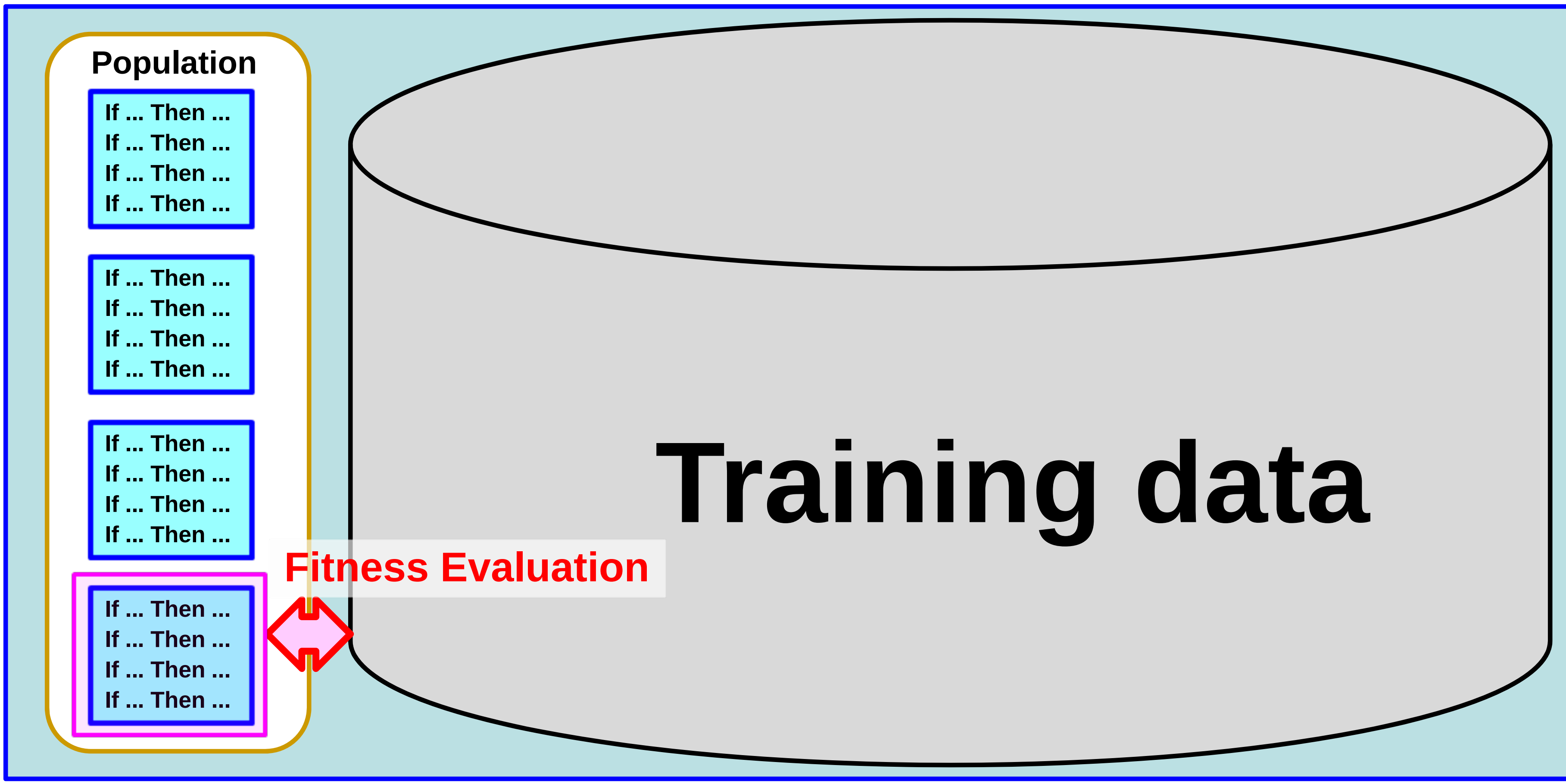


Individual = Rule-Based System (If ... Then ... If ... Then ... If ... Then ... If ... Then ... **)**

Fitness Calculation

Standard Non-Parallel Model

Environment

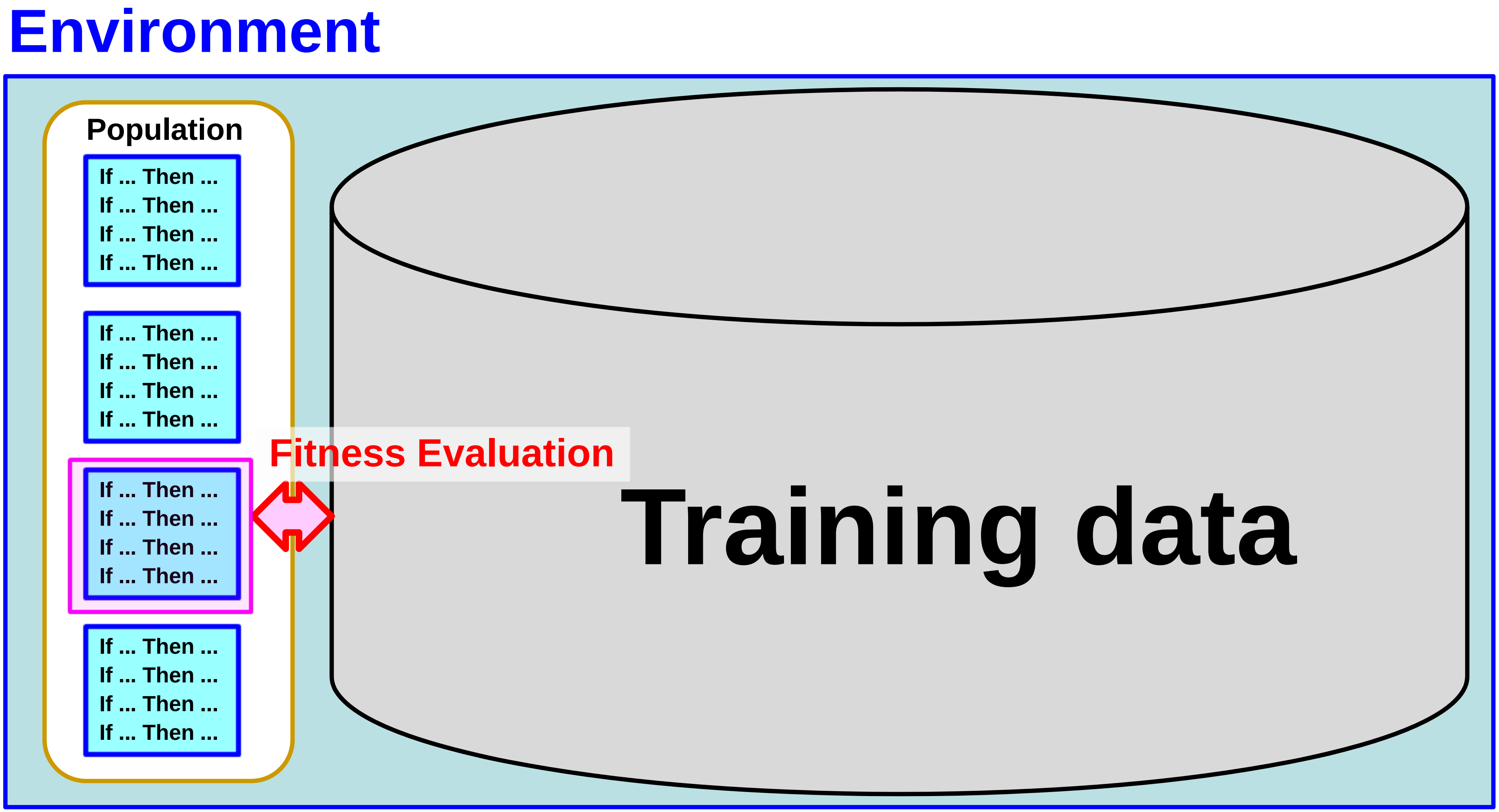


Individual = Rule-Based System (

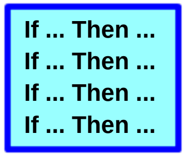
If ... Then ...
If ... Then ...
If ... Then ...
If ... Then ...

)

Fitness Calculation
Standard Non-Parallel Model

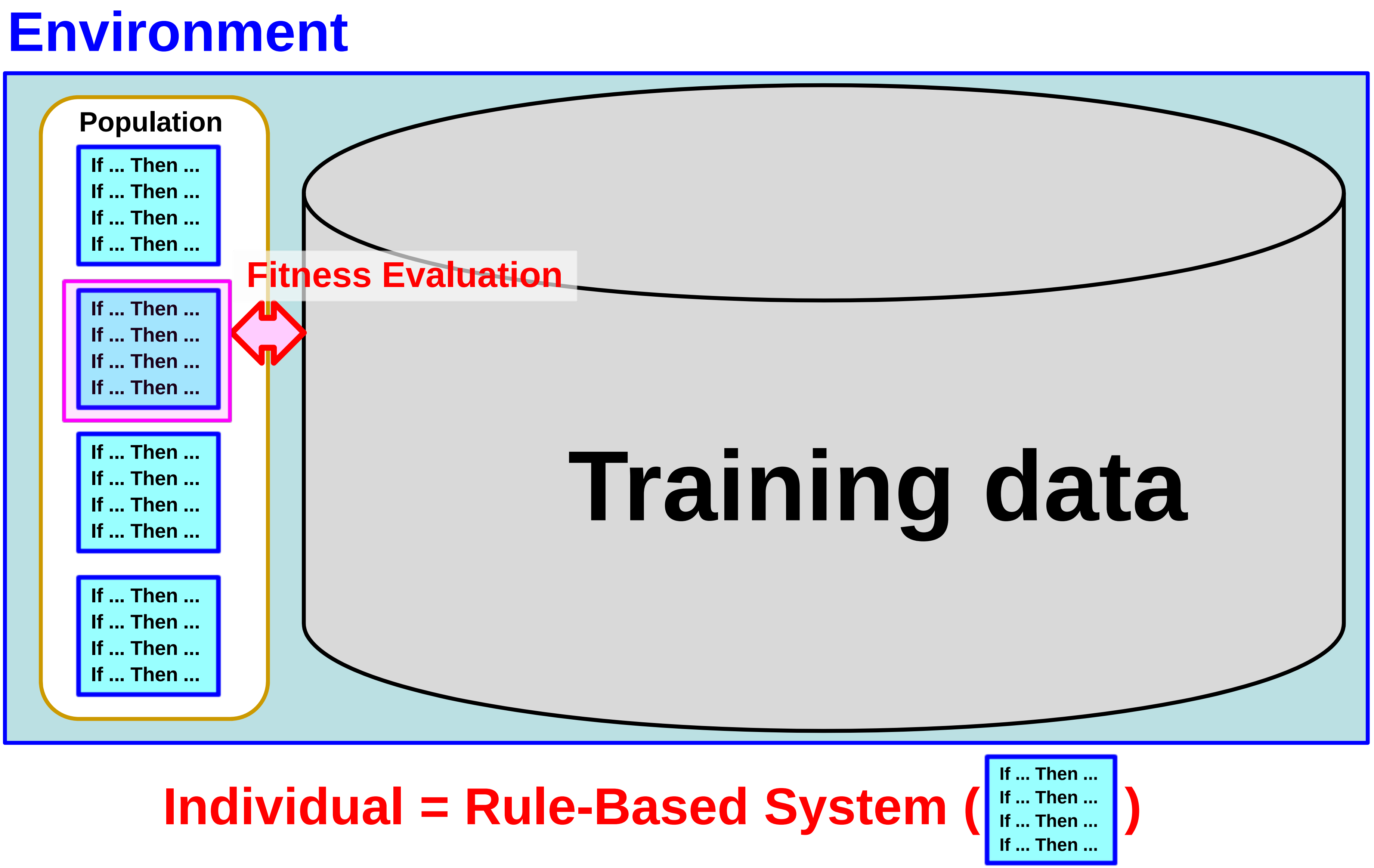


Individual = Rule-Based System (



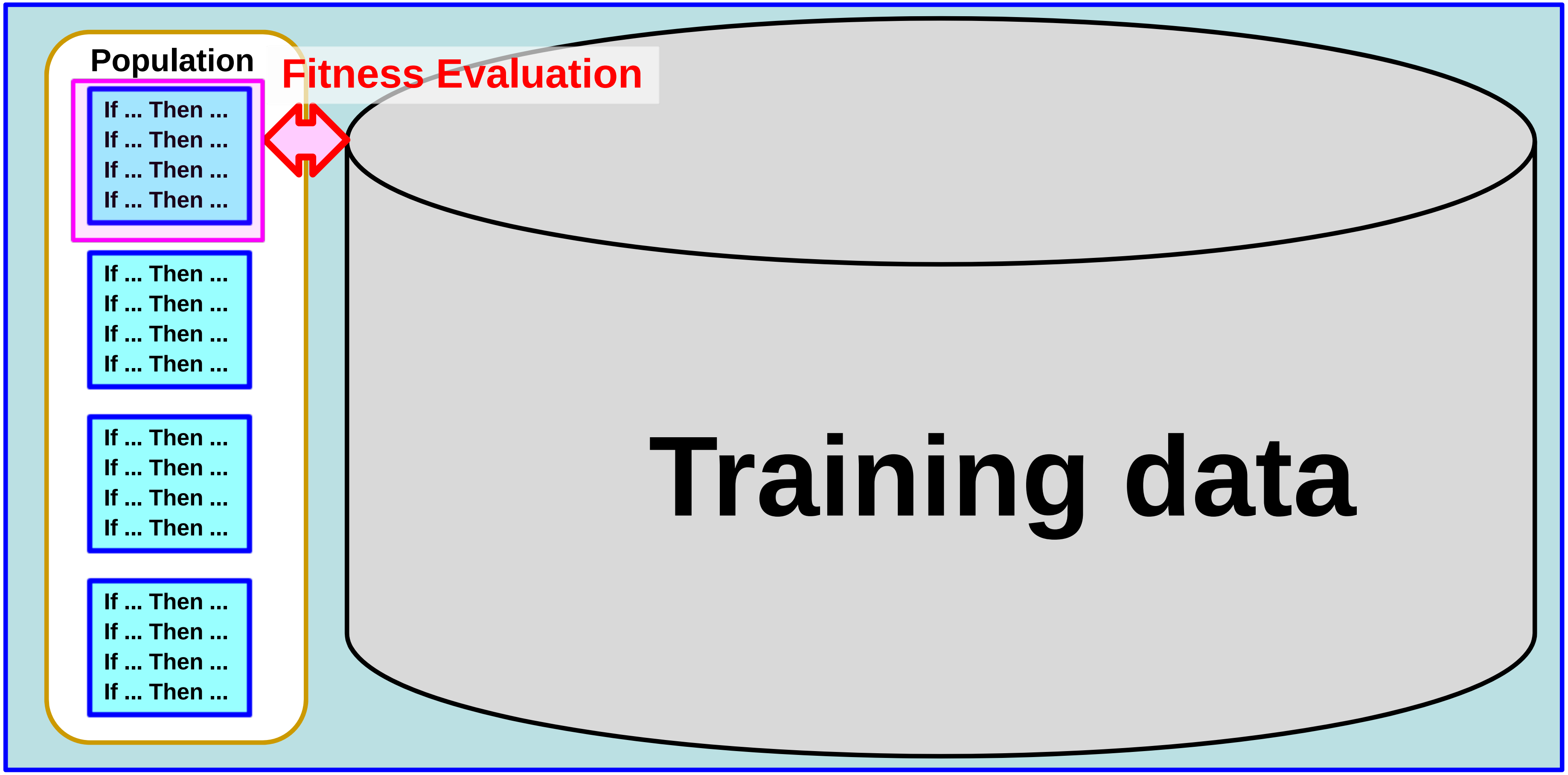
)

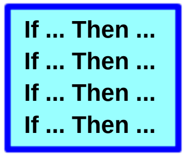
Fitness Calculation
Standard Non-Parallel Model



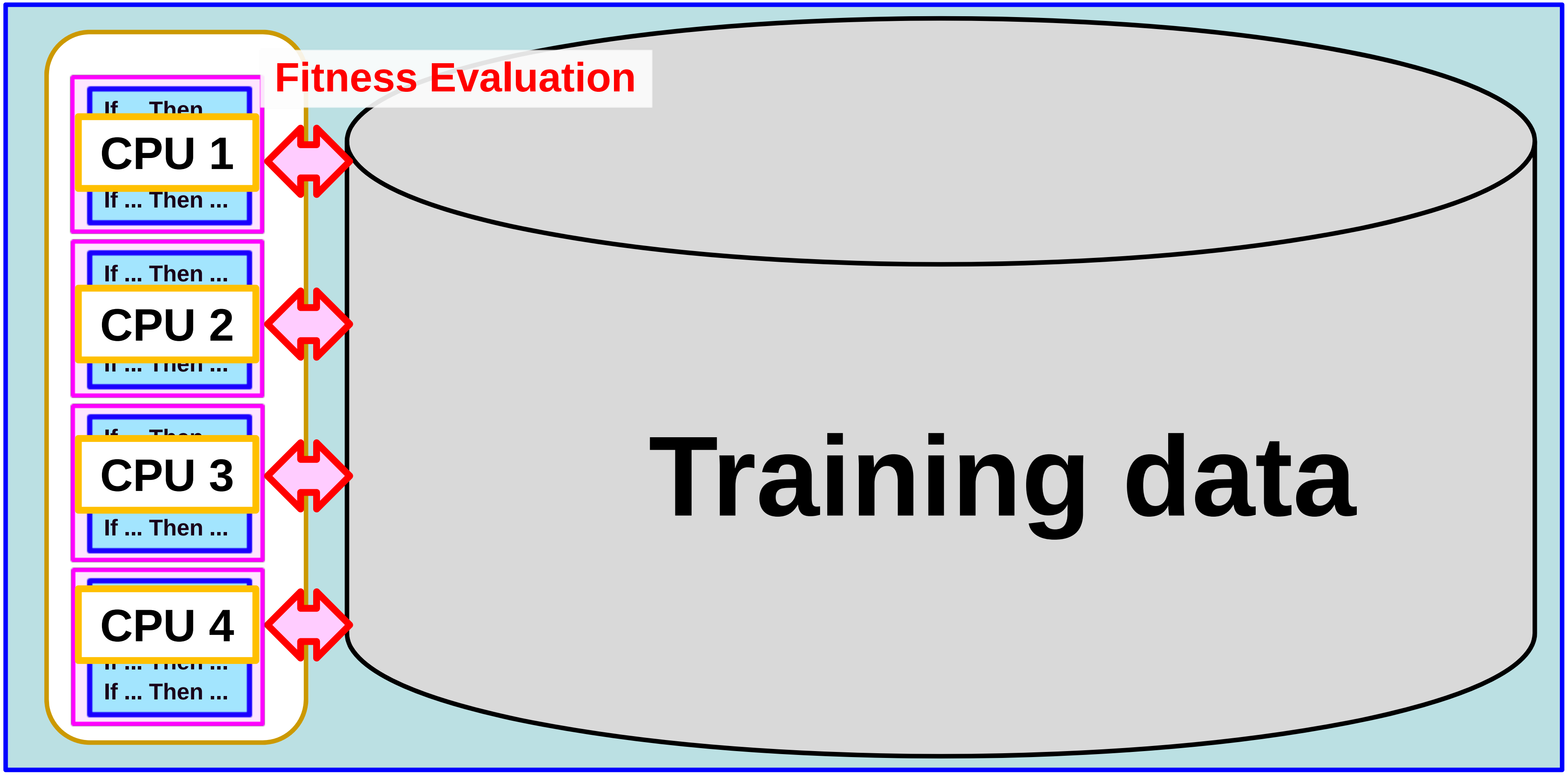
Fitness Calculation
Standard Non-Parallel Model

Environment



Individual = Rule-Based System ( **)**

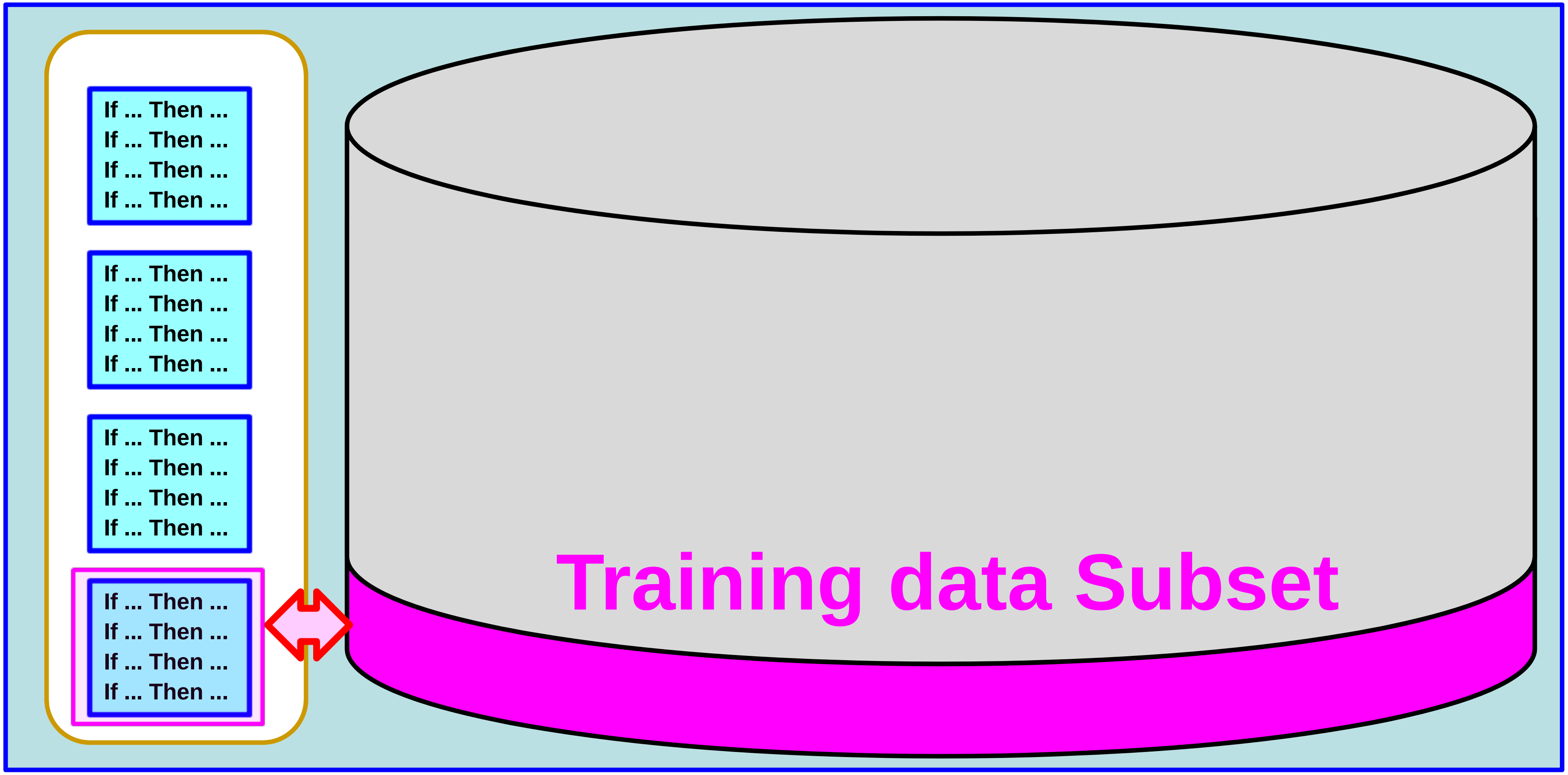
A Popular Approach for Speed-Up
Parallel Computation of Fitness Evaluation



If we use *n* CPUs, the computation load for each CPU can be 1/*n* in comparison with the case of a single CPU (e.g., 25% by four CPUs)

Another Approach for Speed-Up

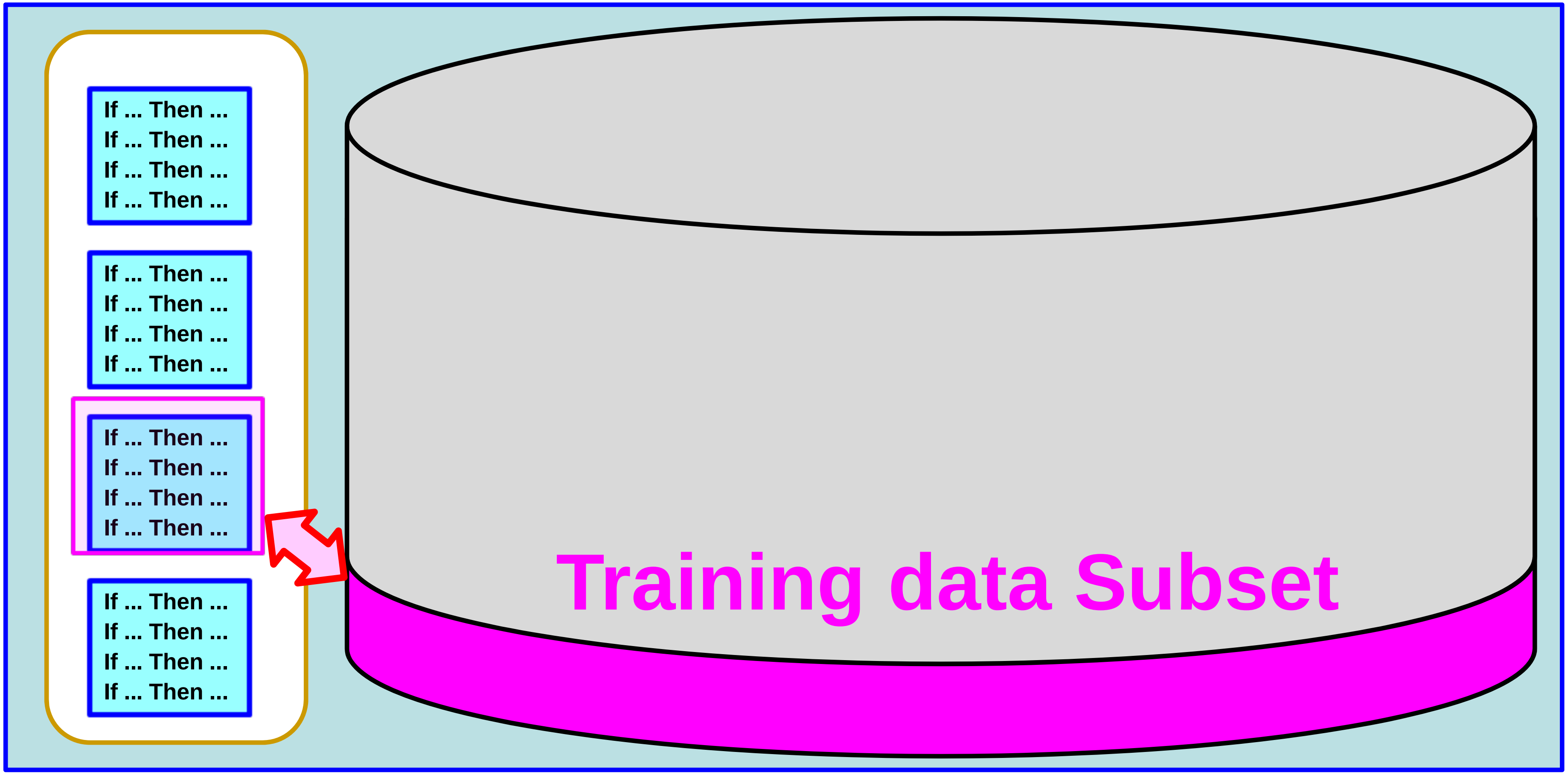
Training Data Reduction



If we use *x*% of the training data, the computation load can be reduced to *x*% in comparison with the use of all the training data.

Another Approach for Speed-Up

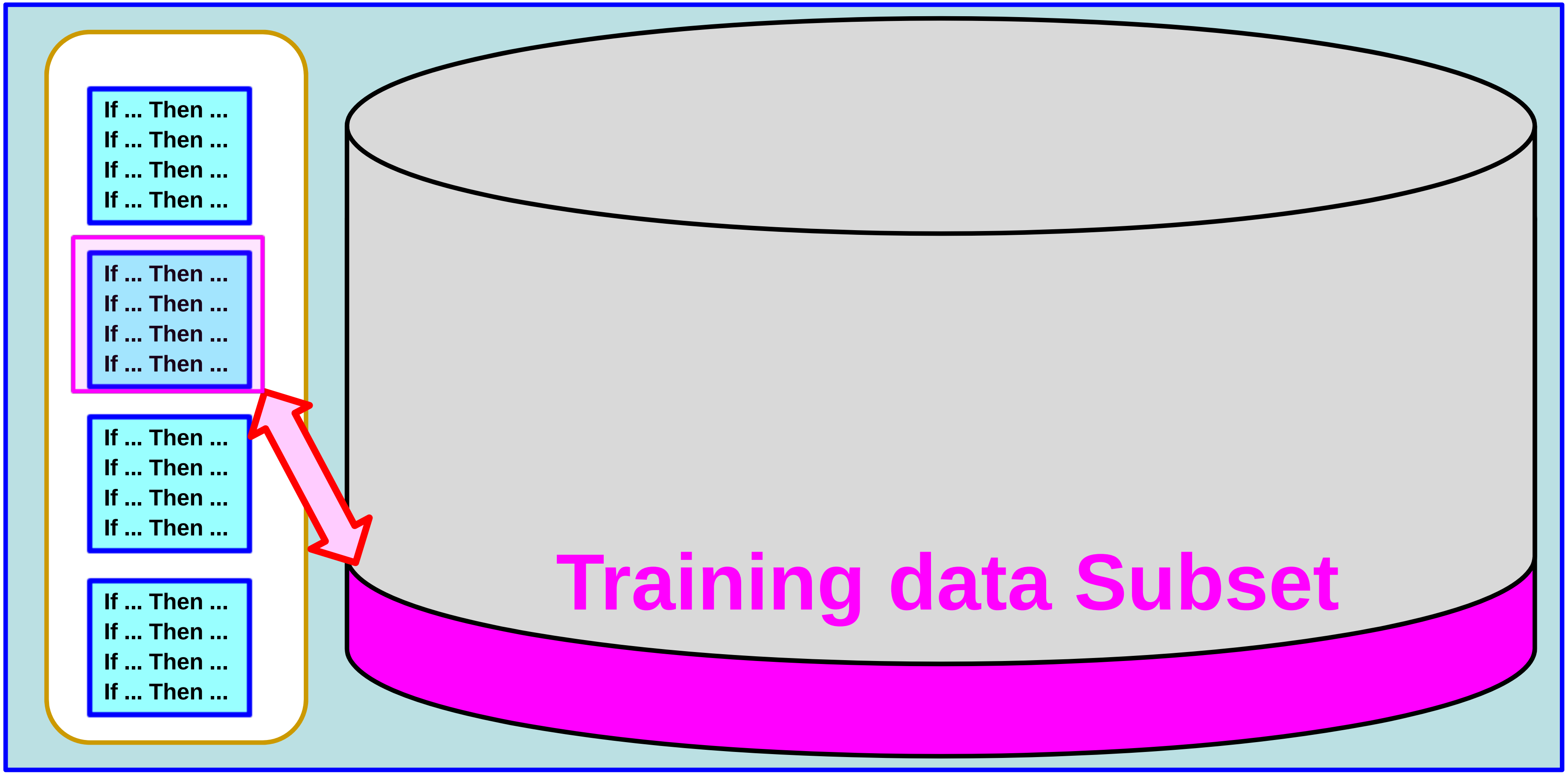
Training Data Reduction



If we use *x*% of the training data, the computation load can be reduced to *x*% in comparison with the use of all the training data.

Another Approach for Speed-Up

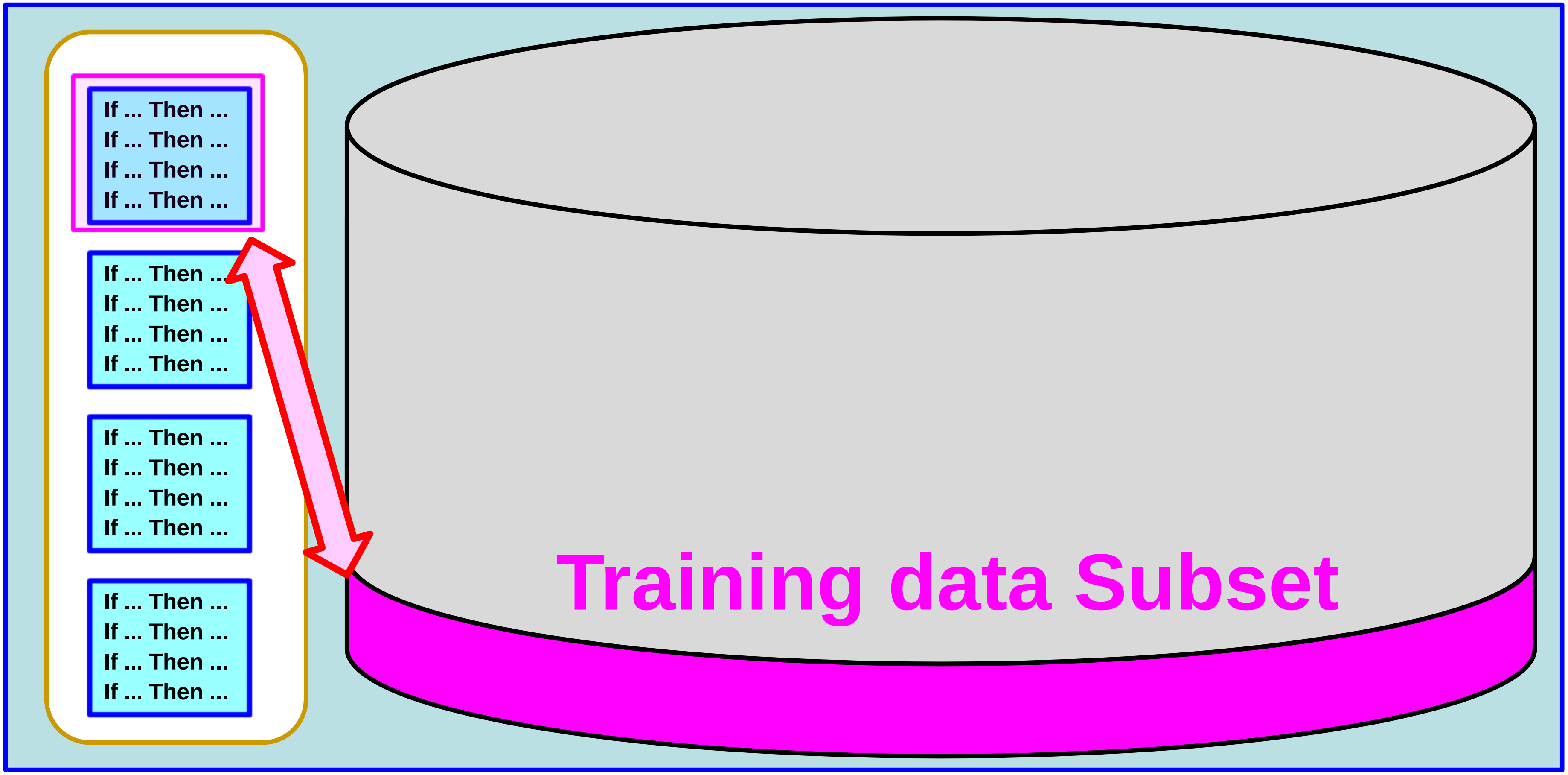
Training Data Reduction



If we use *x*% of the training data, the computation load can be reduced to *x*% in comparison with the use of all the training data.

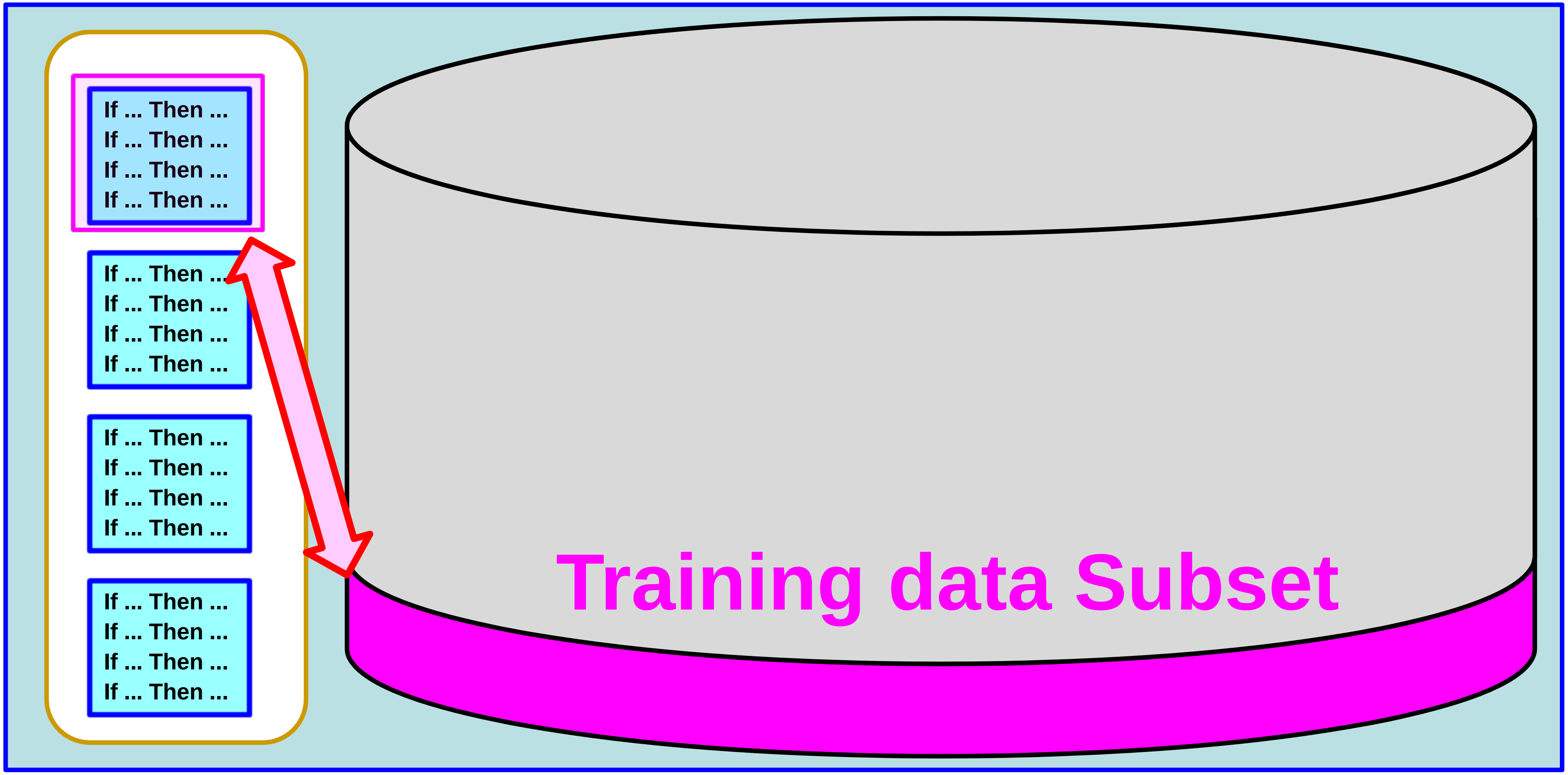
Another Approach for Speed-Up

Training Data Reduction



If we use *x*% of the training data, the computation load can be reduced to *x*% in comparison with the use of all the training data.

Another Approach for Speed-Up
Training Data Reduction

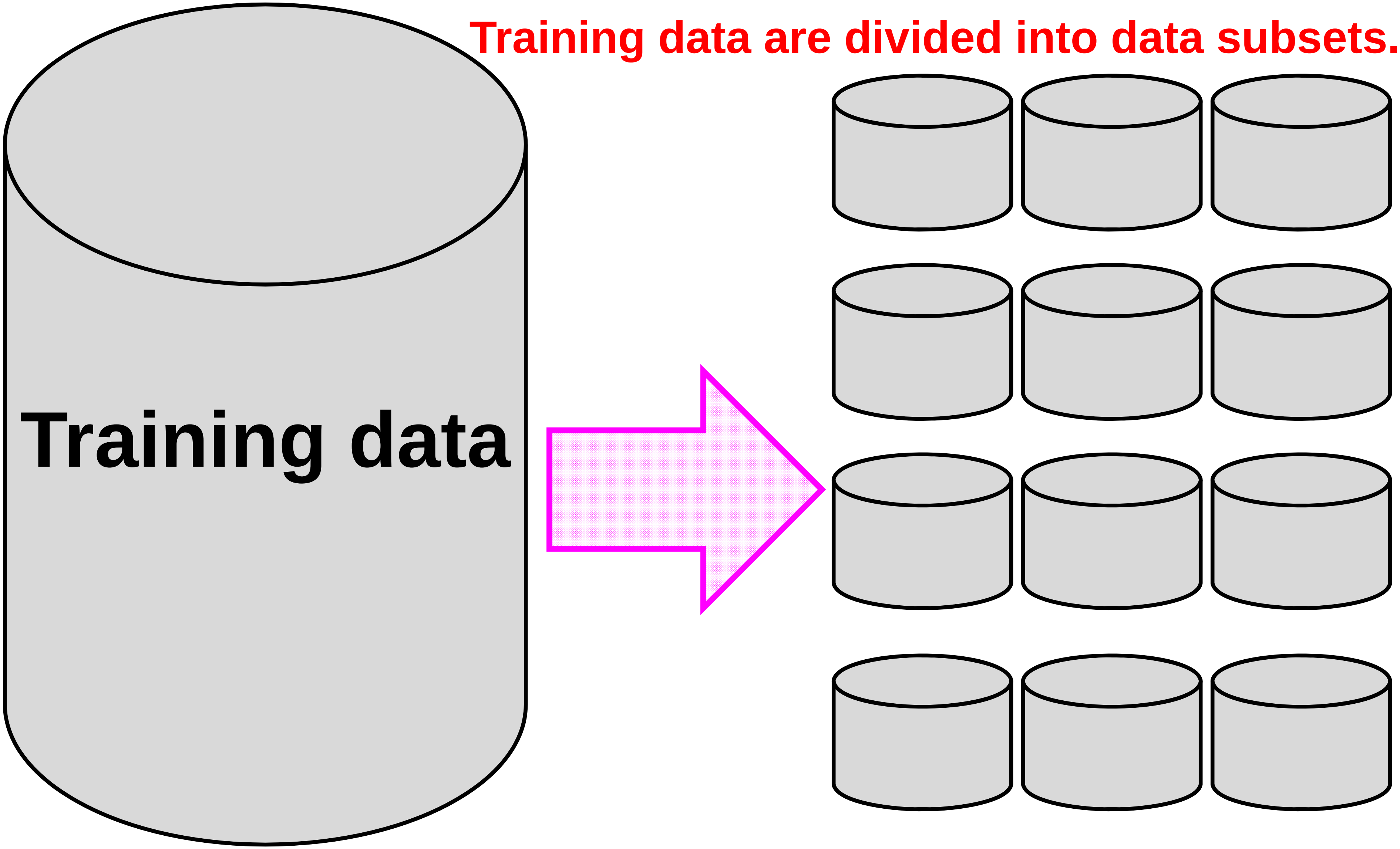


Difficulty: How to choose a training data subset

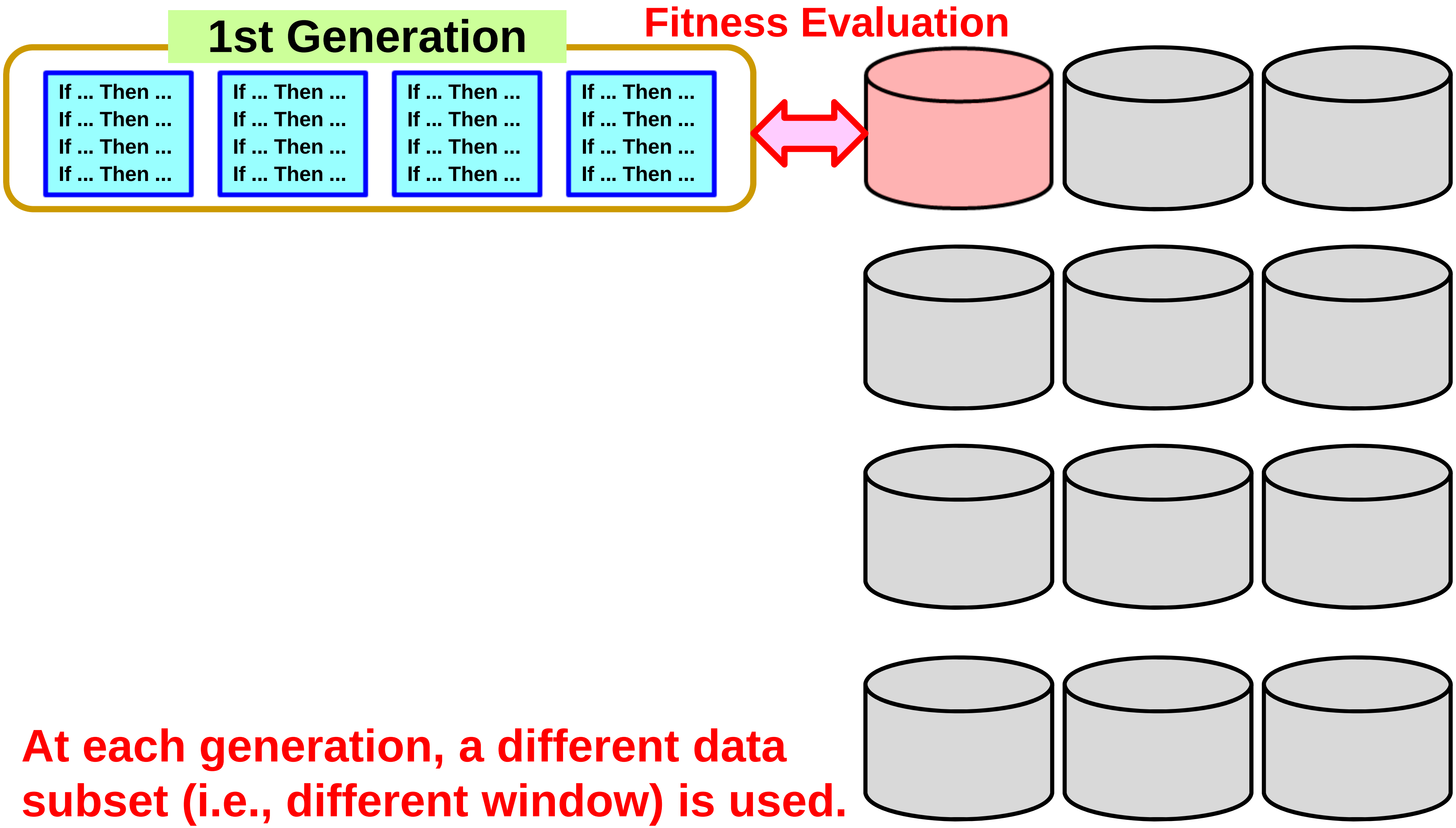
The population will overfit to the selected training data subset.

Idea of Windowing in J. Bacardit et al.: Speeding-up Pittsburgh learning classifier systems: Modeling time and accuracy. PPSN 2004.

Training data are divided into data subsets.

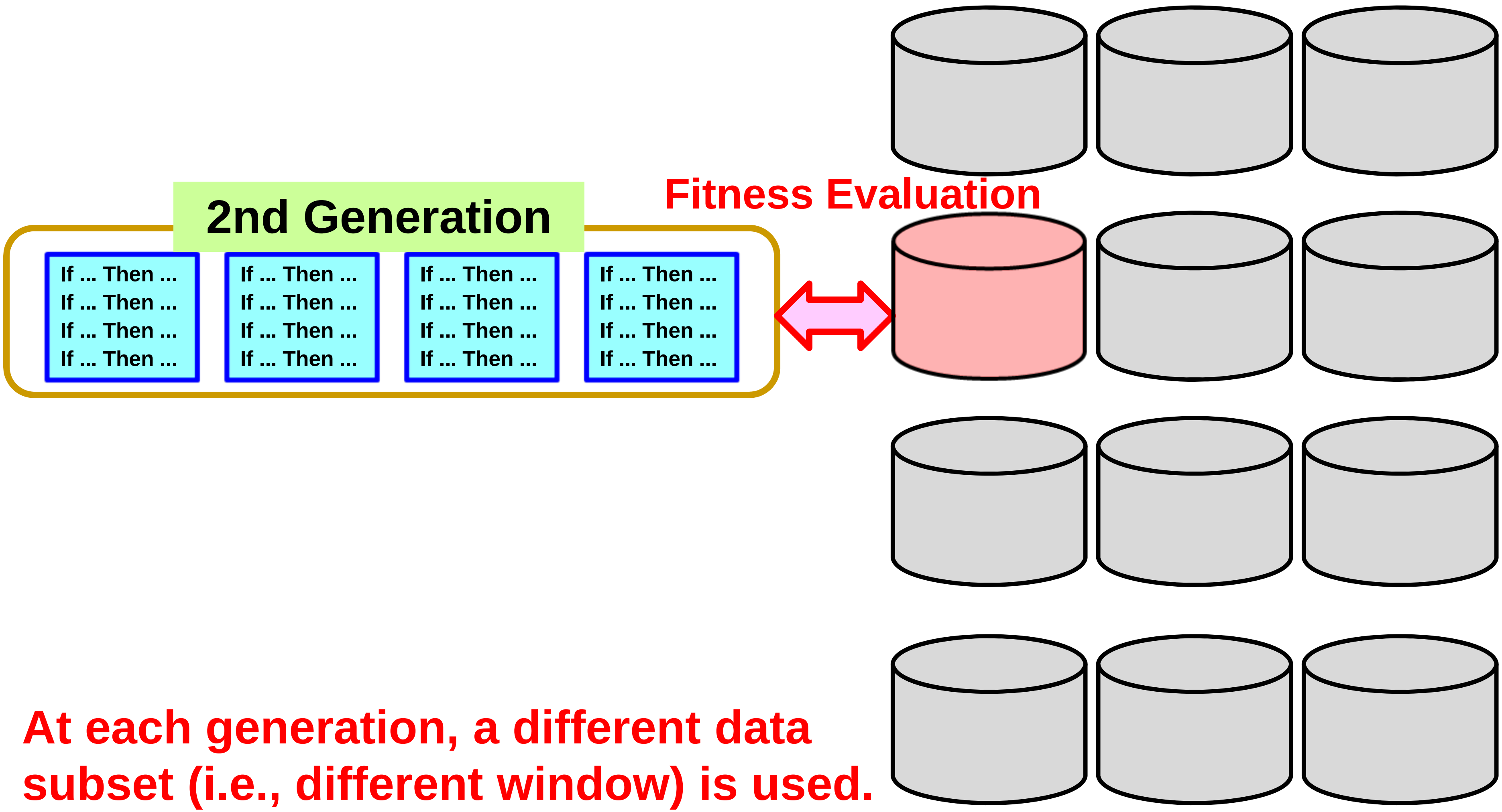


Idea of Windowing in J. Bacardit et al.: Speeding-up Pittsburgh learning classifier systems: Modeling time and accuracy. PPSN 2004.



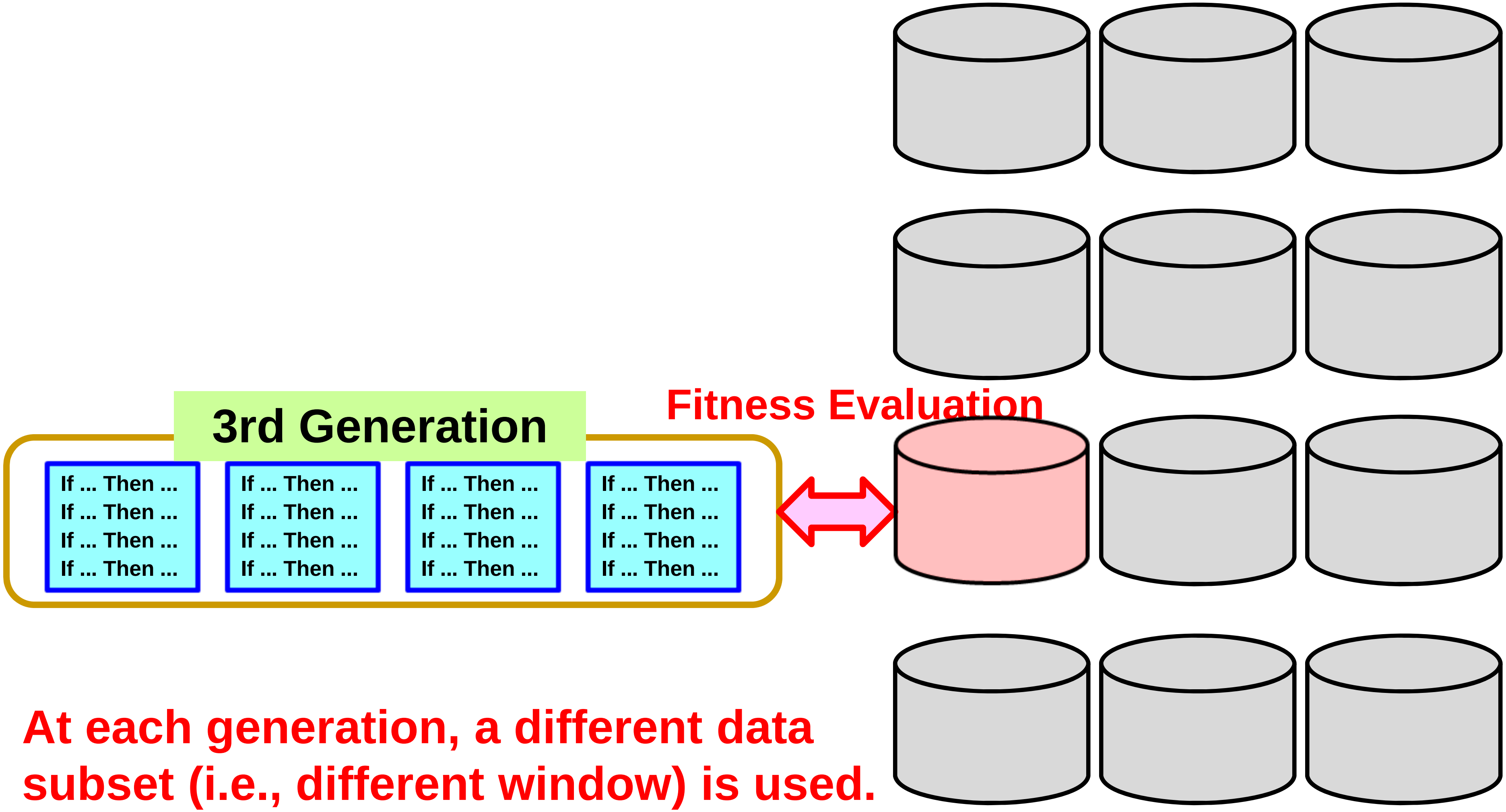
At each generation, a different data subset (i.e., different window) is used.

Idea of Windowing in J. Bacardit et al.: Speeding-up Pittsburgh learning classifier systems: Modeling time and accuracy. PPSN 2004.



At each generation, a different data subset (i.e., different window) is used.

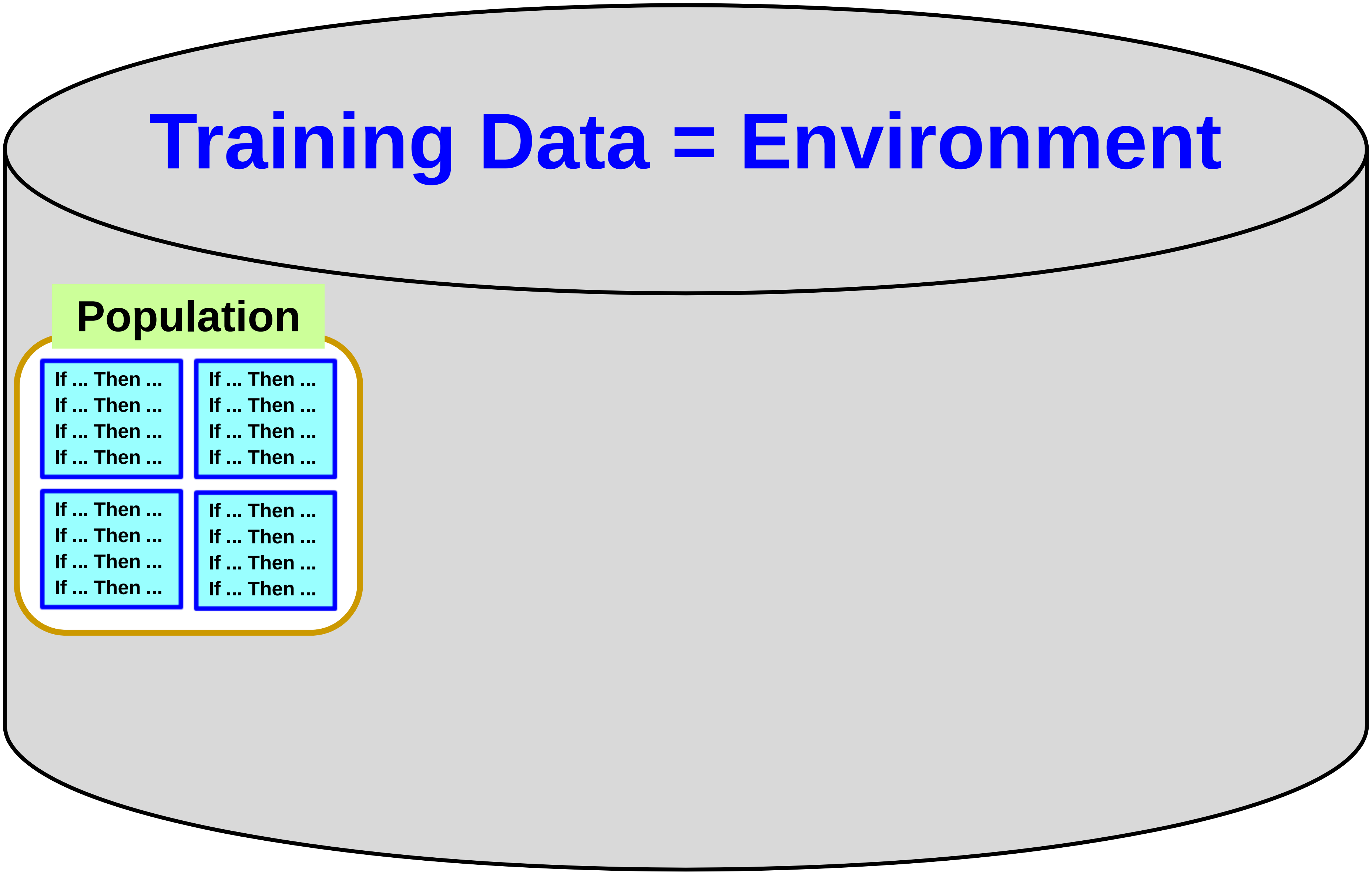
Idea of Windowing in J. Bacardit et al.: Speeding-up Pittsburgh learning classifier systems: Modeling time and accuracy. PPSN 2004.



At each generation, a different data subset (i.e., different window) is used.

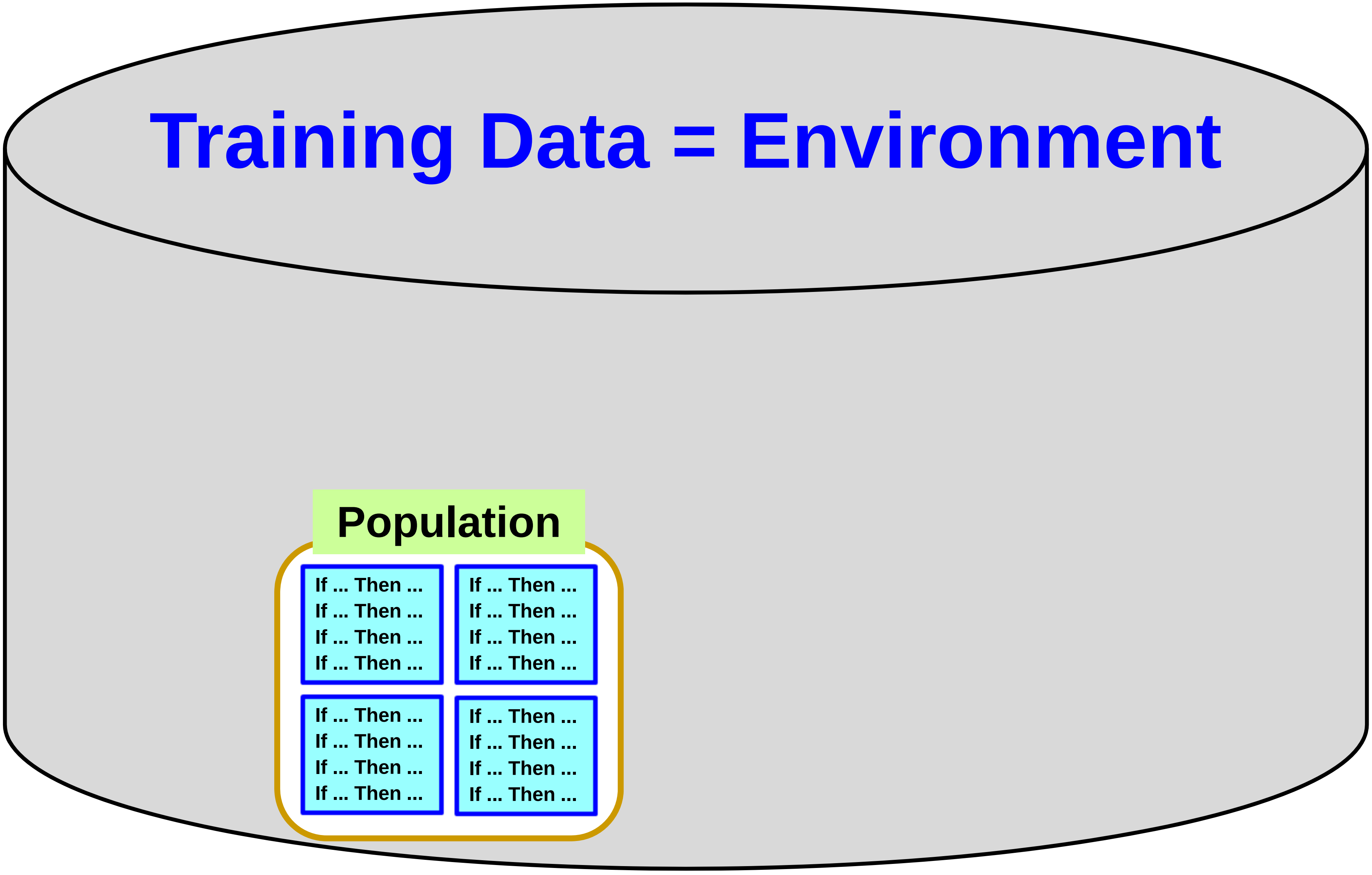
Visual Image of Windowing

A population is moving around in the training data.



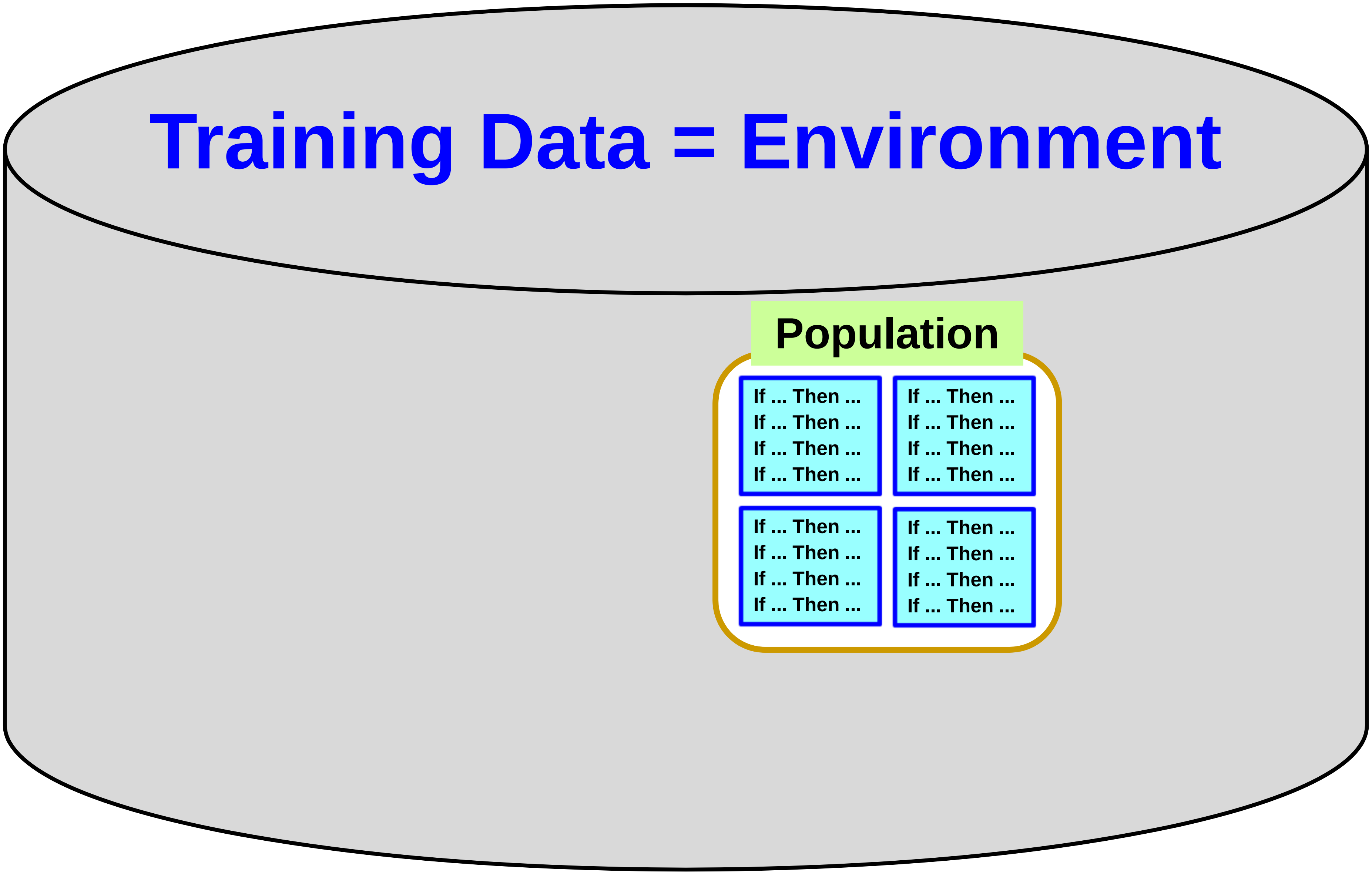
Visual Image of Windowing

A population is moving around in the training data.

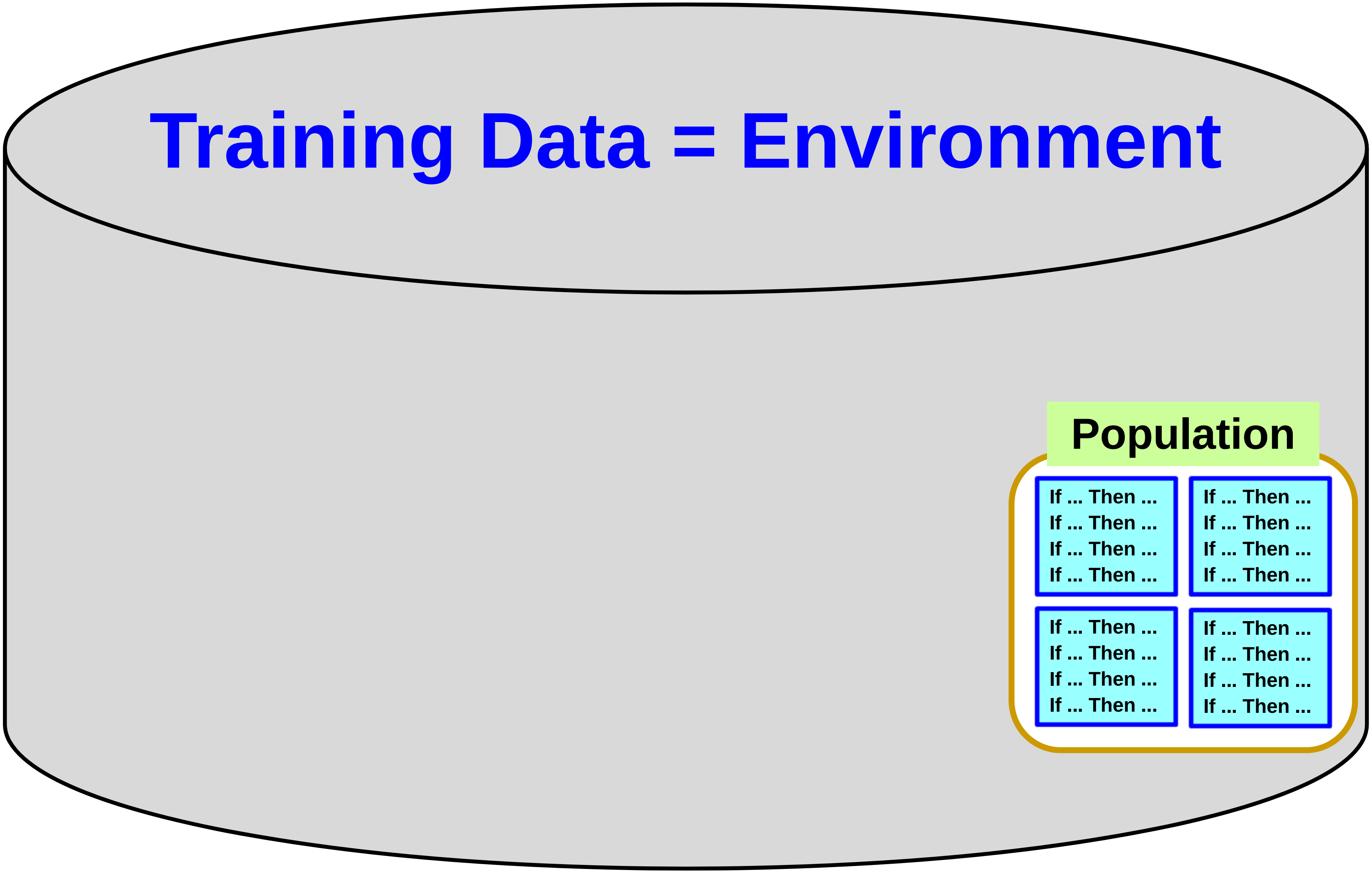


Visual Image of Windowing

A population is moving around in the training data.

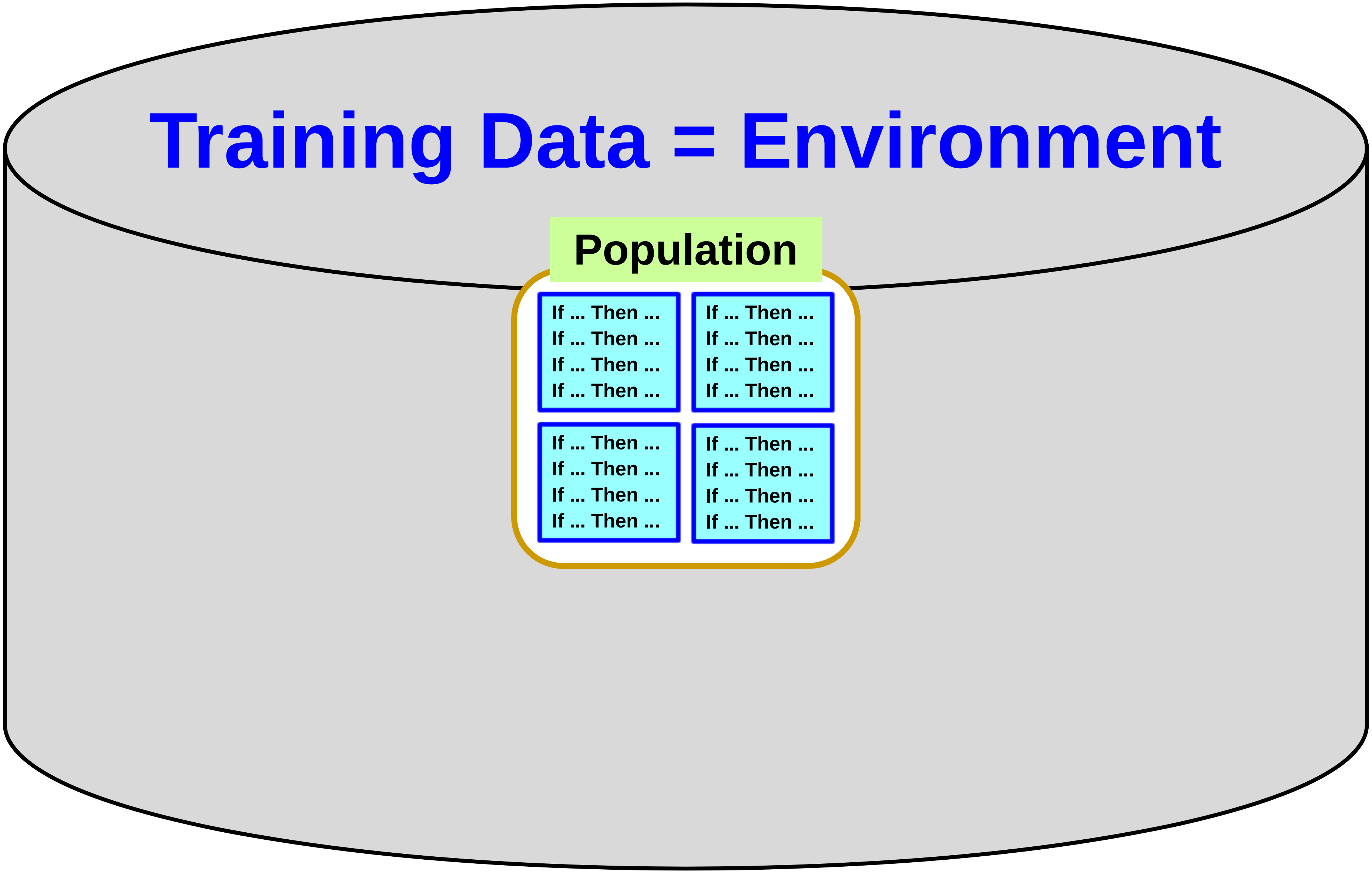


Visual Image of Windowing
A population is moving around in the training data.



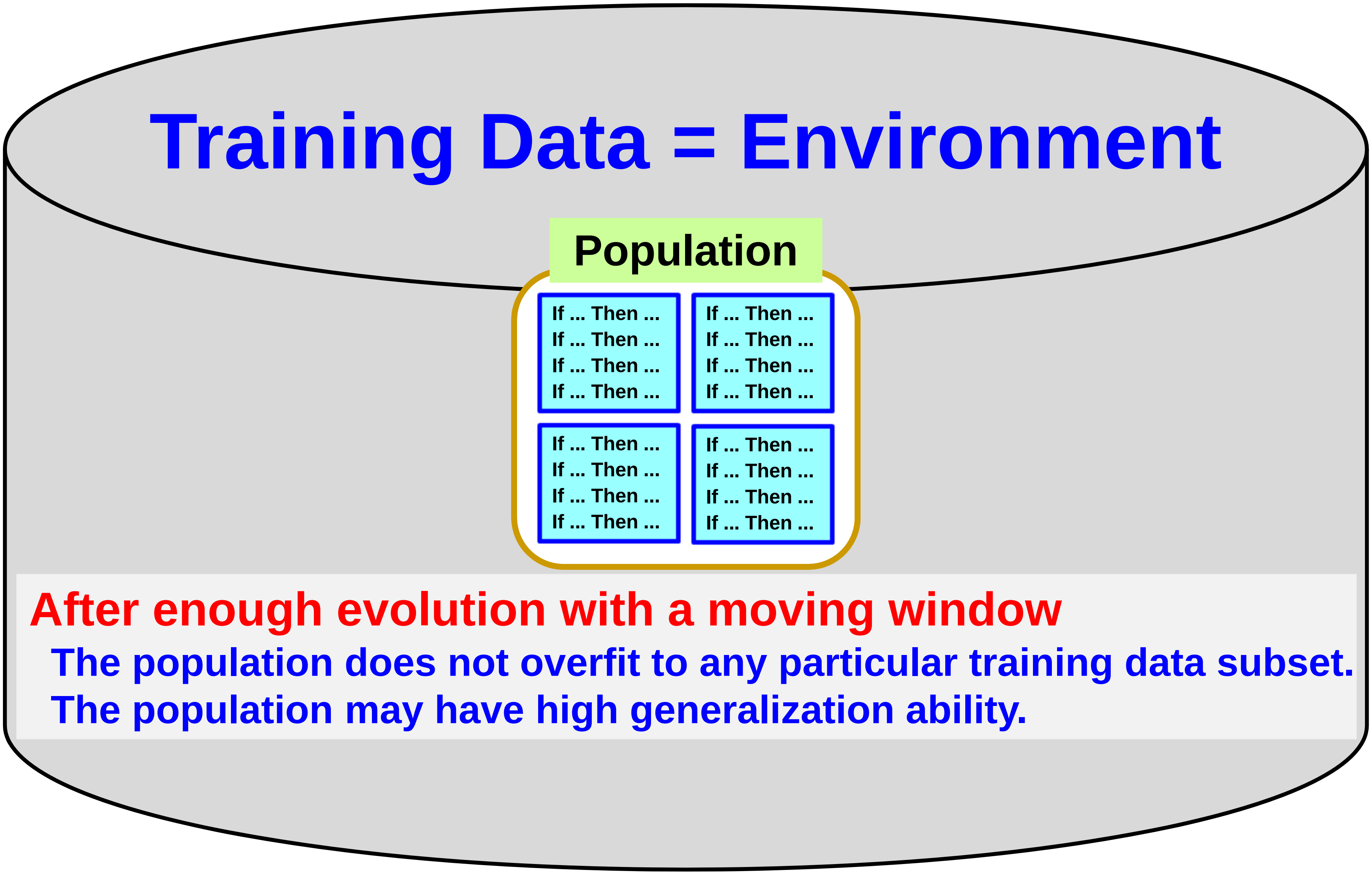
Visual Image of Windowing

A population is moving around in the training data.



Visual Image of Windowing

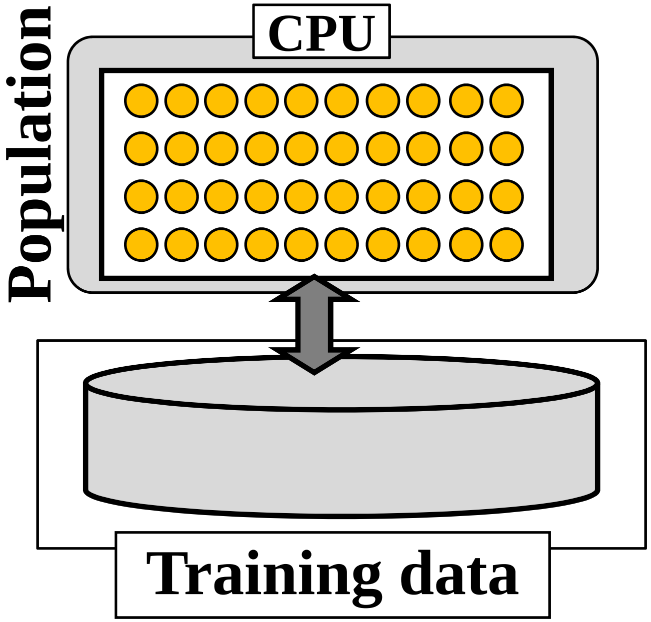
A population is moving around in the training data.



Our Idea: Parallel Distributed Implementation

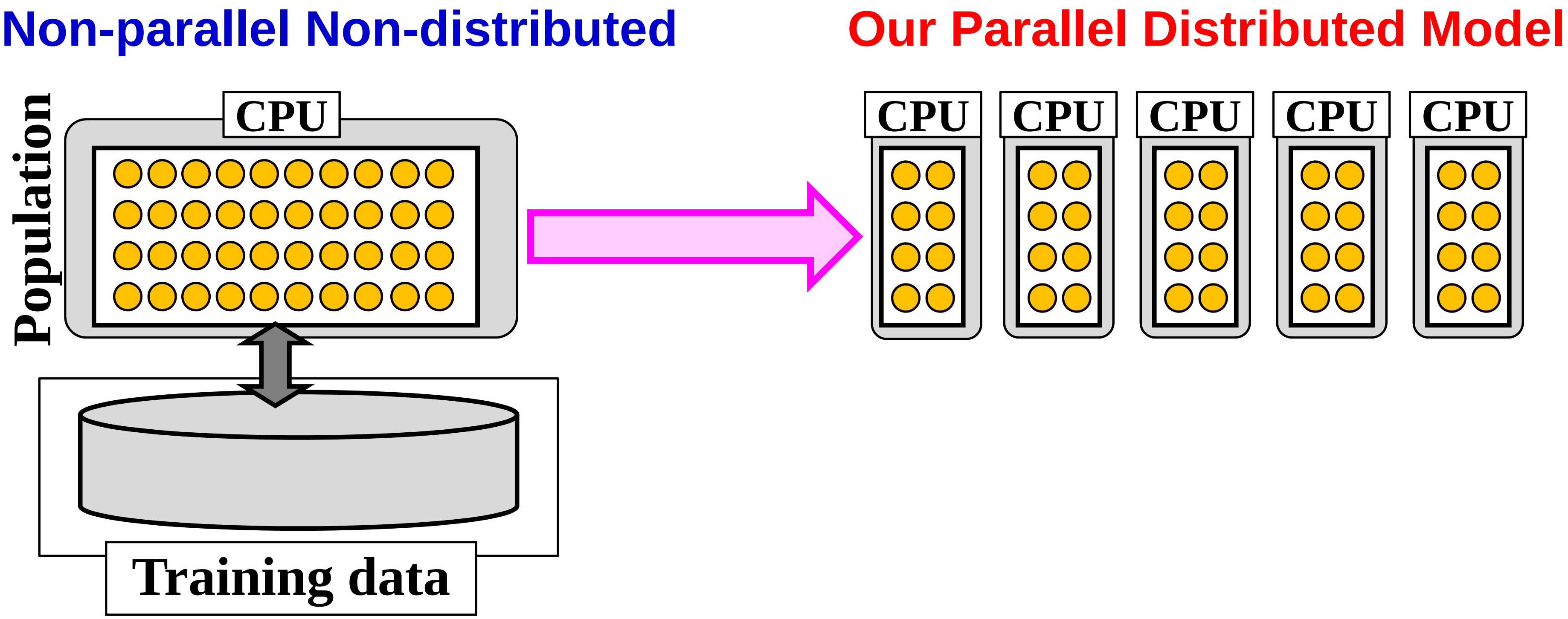
H. Ishibuchi et al.: Parallel Distributed Hybrid Fuzzy GBML Models with Rule Set Migration and Training Data Rotation. TFS (in Press)

Non-parallel Non-distributed



Our Idea: Parallel Distributed Implementation

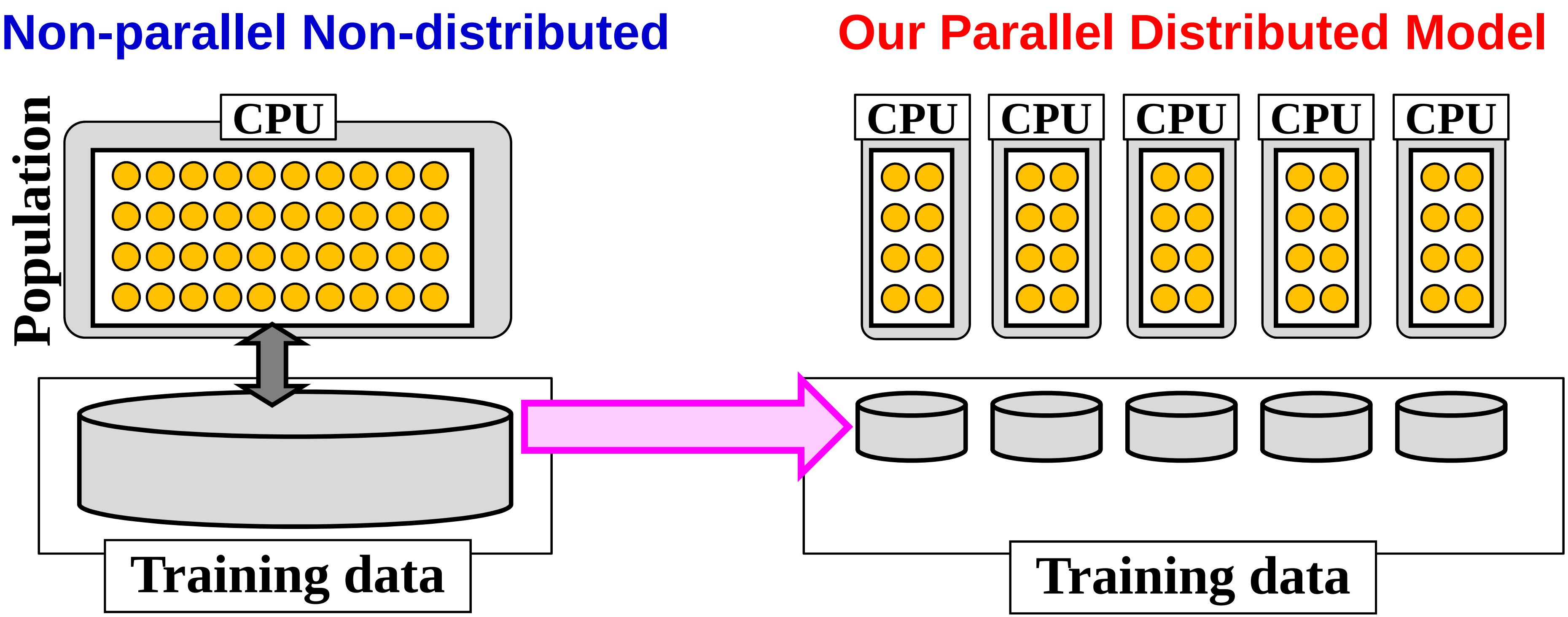
H. Ishibuchi et al.: Parallel Distributed Hybrid Fuzzy GBML Models with Rule Set Migration and Training Data Rotation. TFS (in Press)



(1) A population is divided into multiple subpopulations.
(as in an island model)

Our Idea: Parallel Distributed Implementation

H. Ishibuchi et al.: Parallel Distributed Hybrid Fuzzy GBML Models with Rule Set Migration and Training Data Rotation. TFS (in Press)

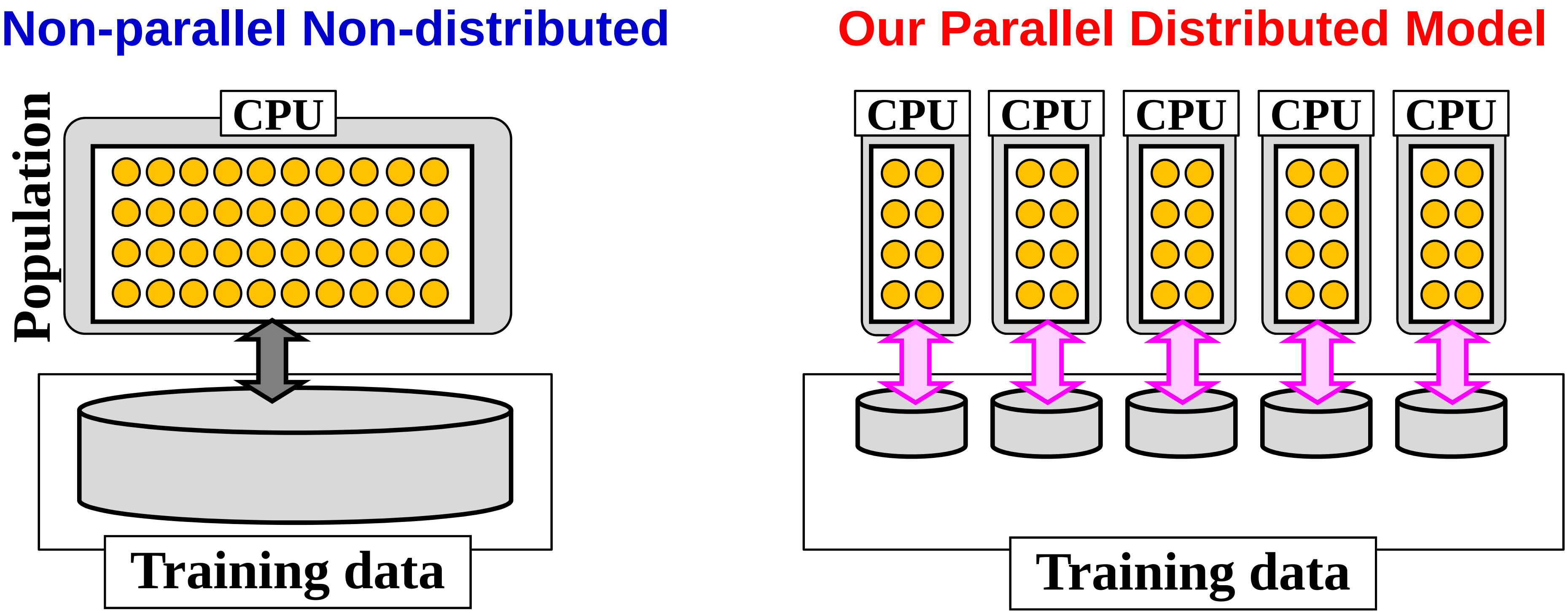


(1) A population is divided into multiple subpopulations.

(2) Training data are also divided into multiple subsets. (as in the windowing method)

Our Idea: Parallel Distributed Implementation

H. Ishibuchi et al.: Parallel Distributed Hybrid Fuzzy GBML Models with Rule Set Migration and Training Data Rotation. TFS (in Press)



(1) A population is divided into multiple subpopulations.

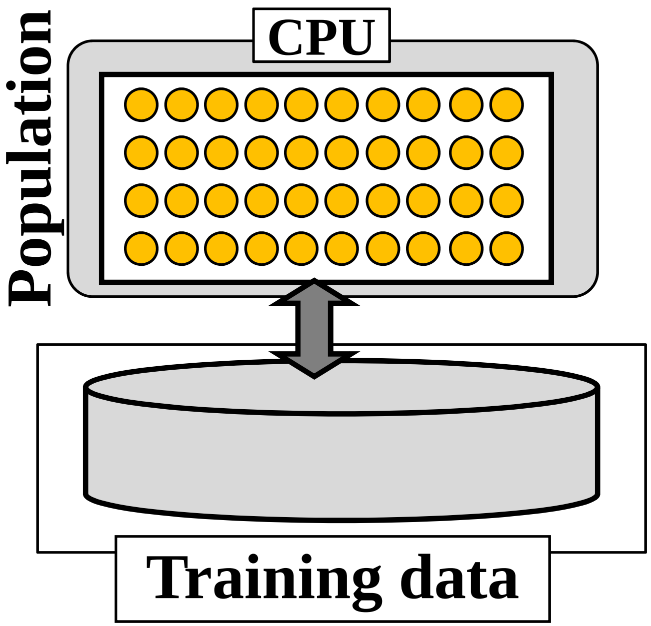
(2) Training data are also divided into multiple subsets.

(3) An evolutionary algorithm is locally performed at each CPU. (as in an island model)

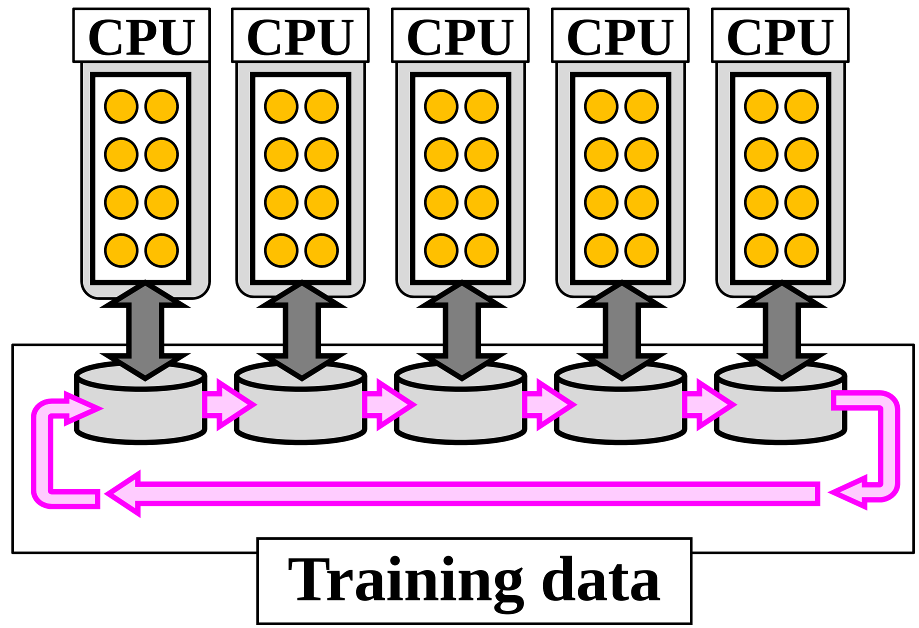
Our Idea: Parallel Distributed Implementation

H. Ishibuchi et al.: Parallel Distributed Hybrid Fuzzy GBML Models with Rule Set Migration and Training Data Rotation. TFS (in Press)

Non-parallel Non-distributed



Our Parallel Distributed Model

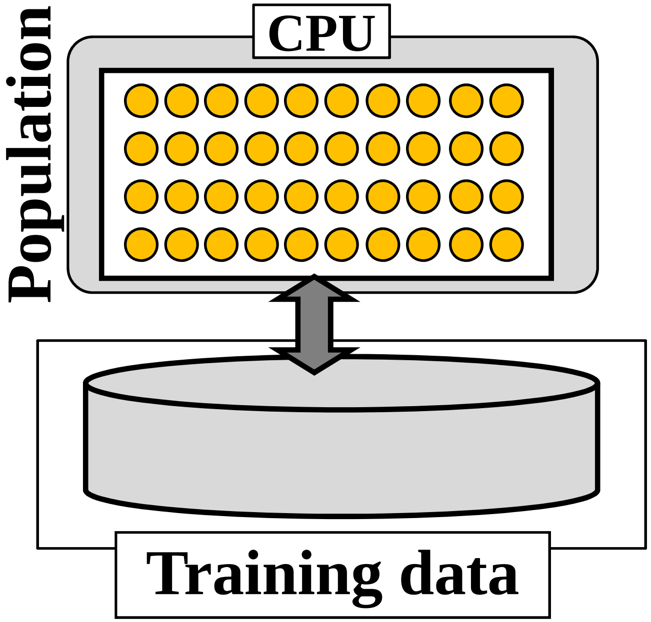


(1) A population is divided into multiple subpopulations.
(2) Training data are also divided into multiple subsets.
(3) An evolutionary algorithm is locally performed at each CPU.
(4) Training data subsets are periodically rotated.
(e.g., every 100 generations)

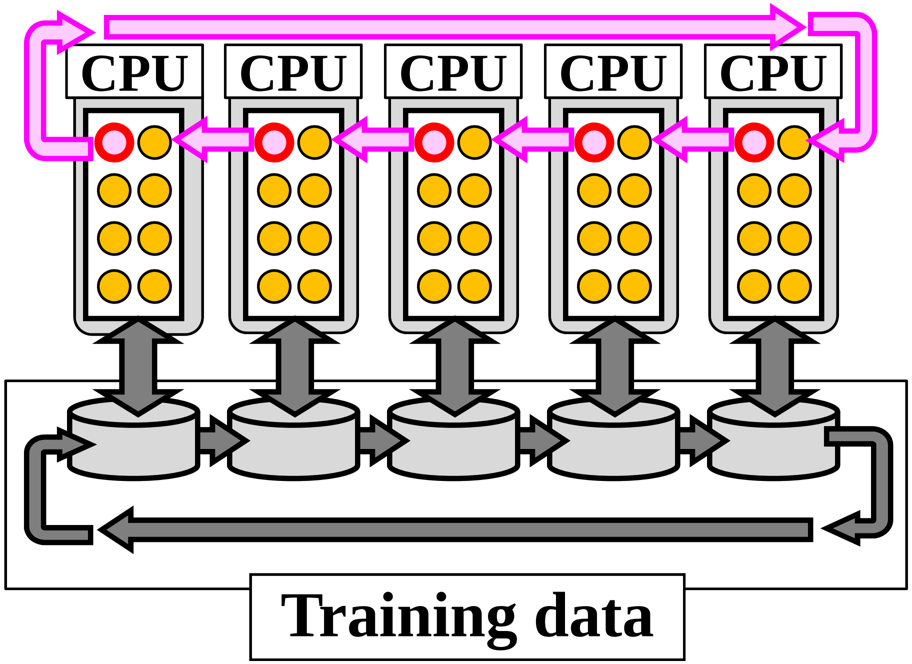
Our Idea: Parallel Distributed Implementation

H. Ishibuchi et al.: Parallel Distributed Hybrid Fuzzy GBML Models with Rule Set Migration and Training Data Rotation. TFS (in Press)

Non-parallel Non-distributed



Our Parallel Distributed Model



(1) A population is divided into multiple subpopulations.

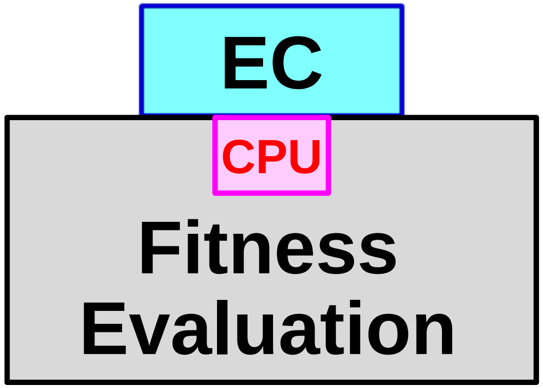
(2) Training data are also divided into multiple subsets.

(3) An evolutionary algorithm is locally performed at each CPU.

(4) Training data subsets are periodically rotated.

(5) Migration is also periodically performed.

Computation Load of Four Models

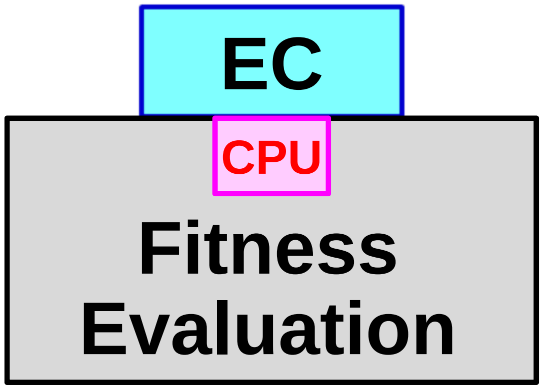


Standard Non-Parallel Model

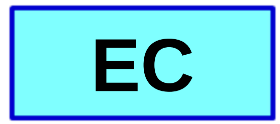
Computation Load

= EC Part + Fitness Evaluation Part

Computation Load of Four Models



EC = Evolutionary Computation



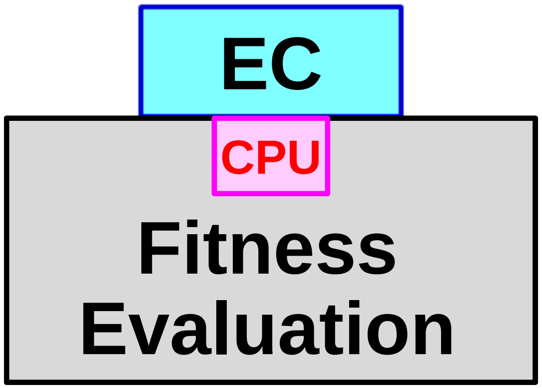
= { Selection, Crossover,
Mutation, Generation Update }

Standard Non-Parallel Model

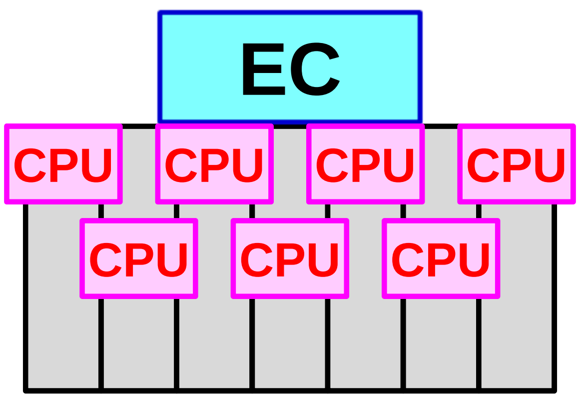
Computation Load

= EC Part + Fitness Evaluation Part

Computation Load of Four Models



Standard Non-Parallel Model



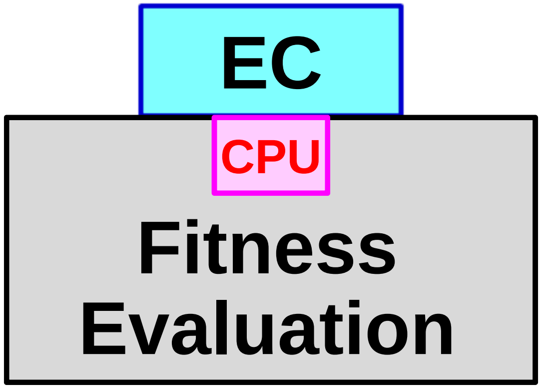
Standard Parallel Model
(Parallel Fitness Evaluation)

Computation Load

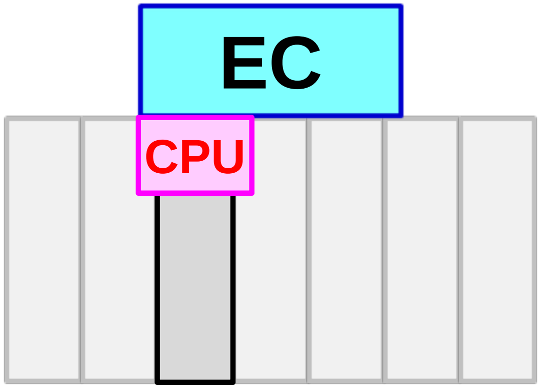
= EC Part

+ Fitness Evaluation Part (1/7)

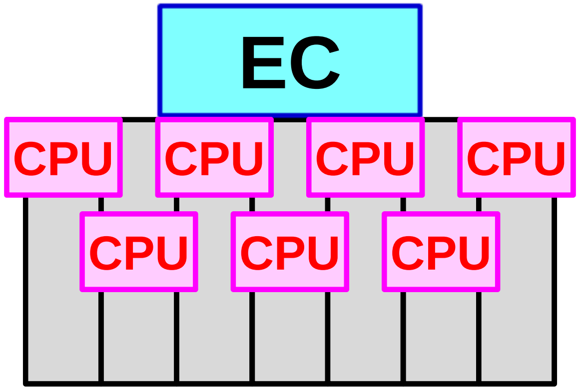
Computation Load of Four Models



Standard Non-Parallel Model



Windowing Model
(Reduced Training Data Set)



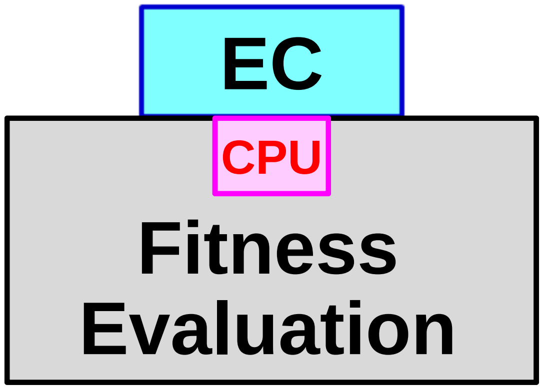
Standard Parallel Model
(Parallel Fitness Evaluation)

Computation Load

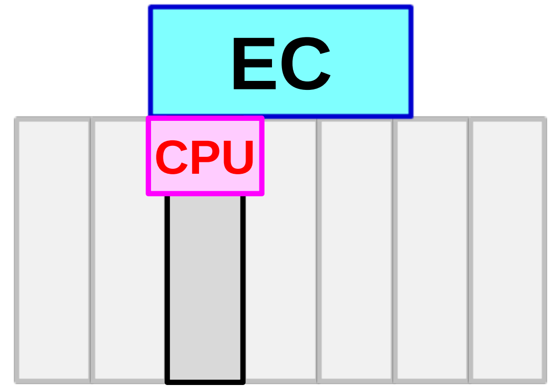
= EC Part

+ Fitness Evaluation Part (1/7)

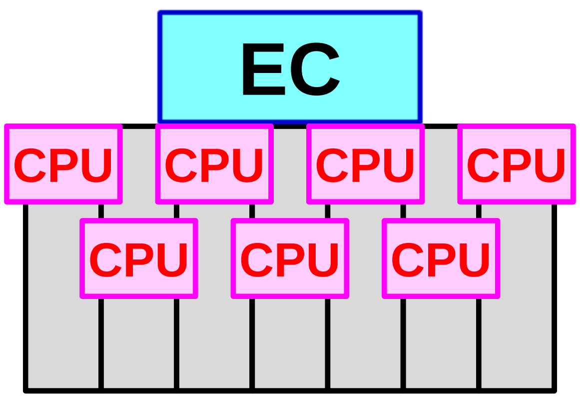
Computation Load of Four Models



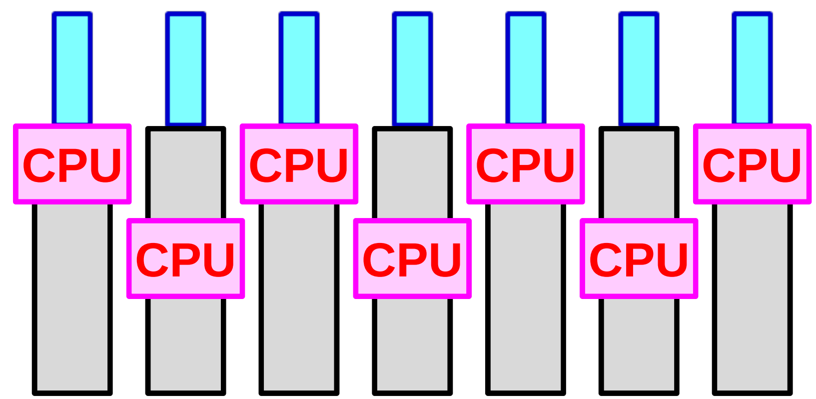
Standard Non-Parallel Model



Windowing Model
(Reduced Training Data Set)

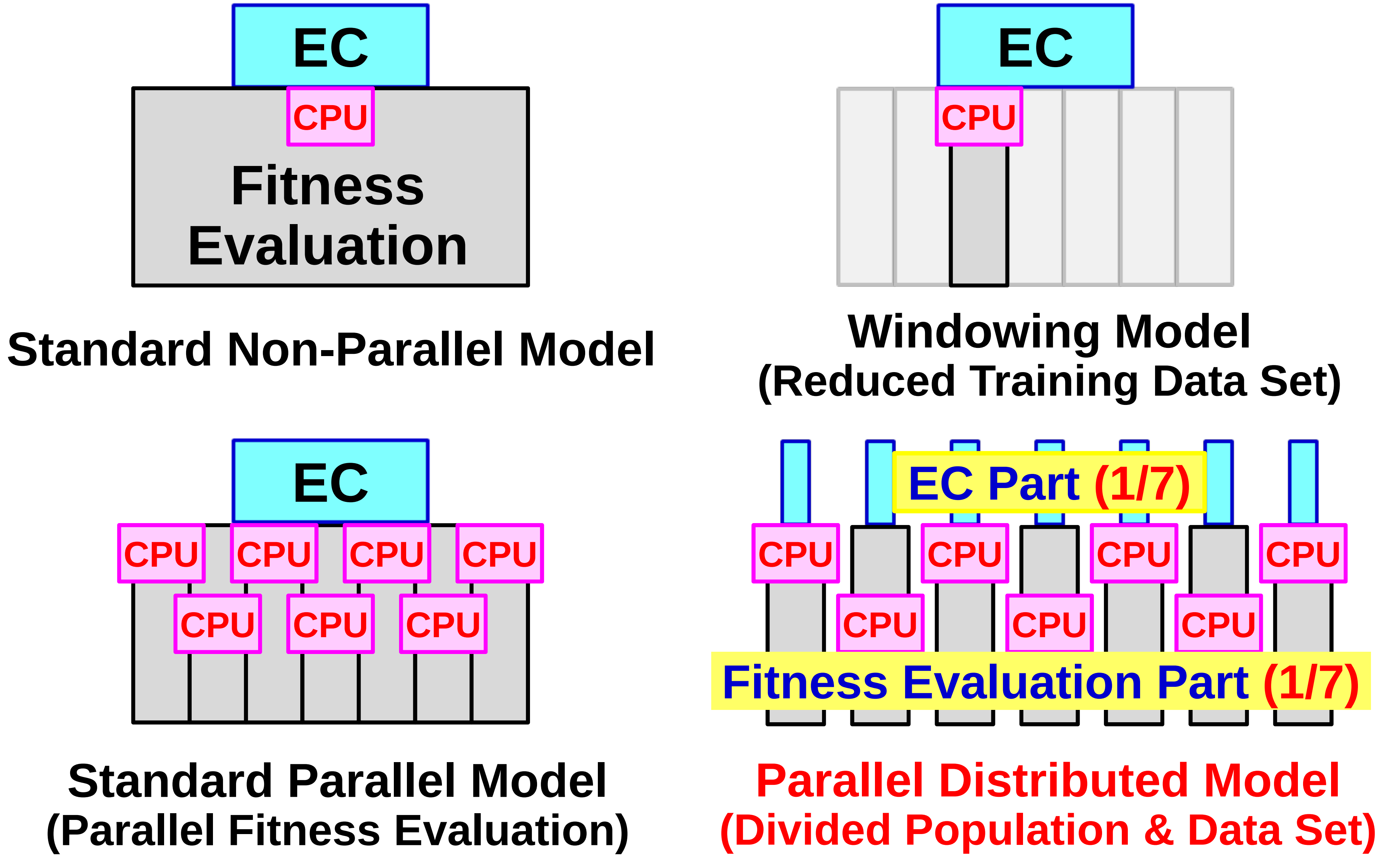


Standard Parallel Model
(Parallel Fitness Evaluation)



Parallel Distributed Model
(Divided Population & Data Set)

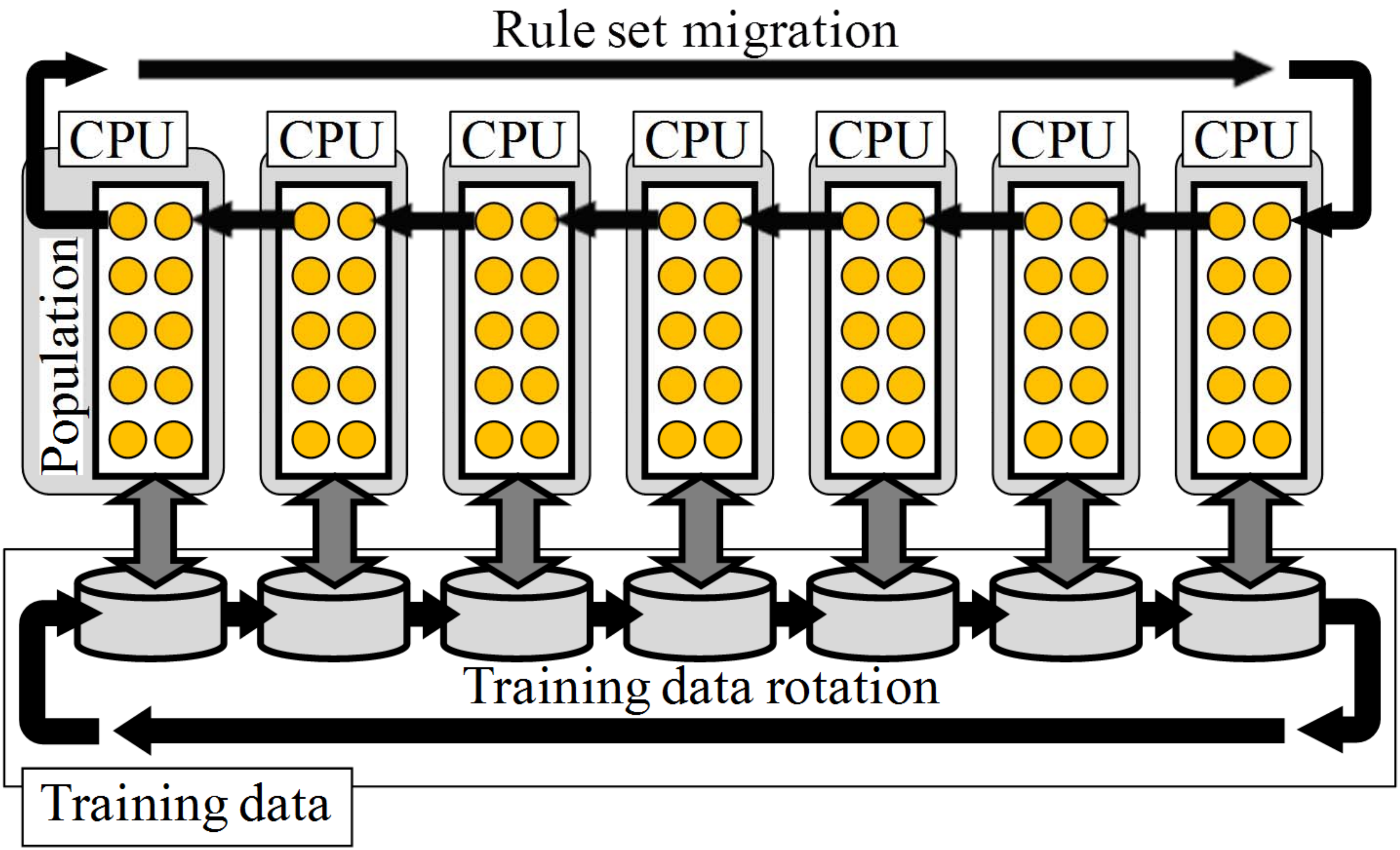
Computation Load of Four Models



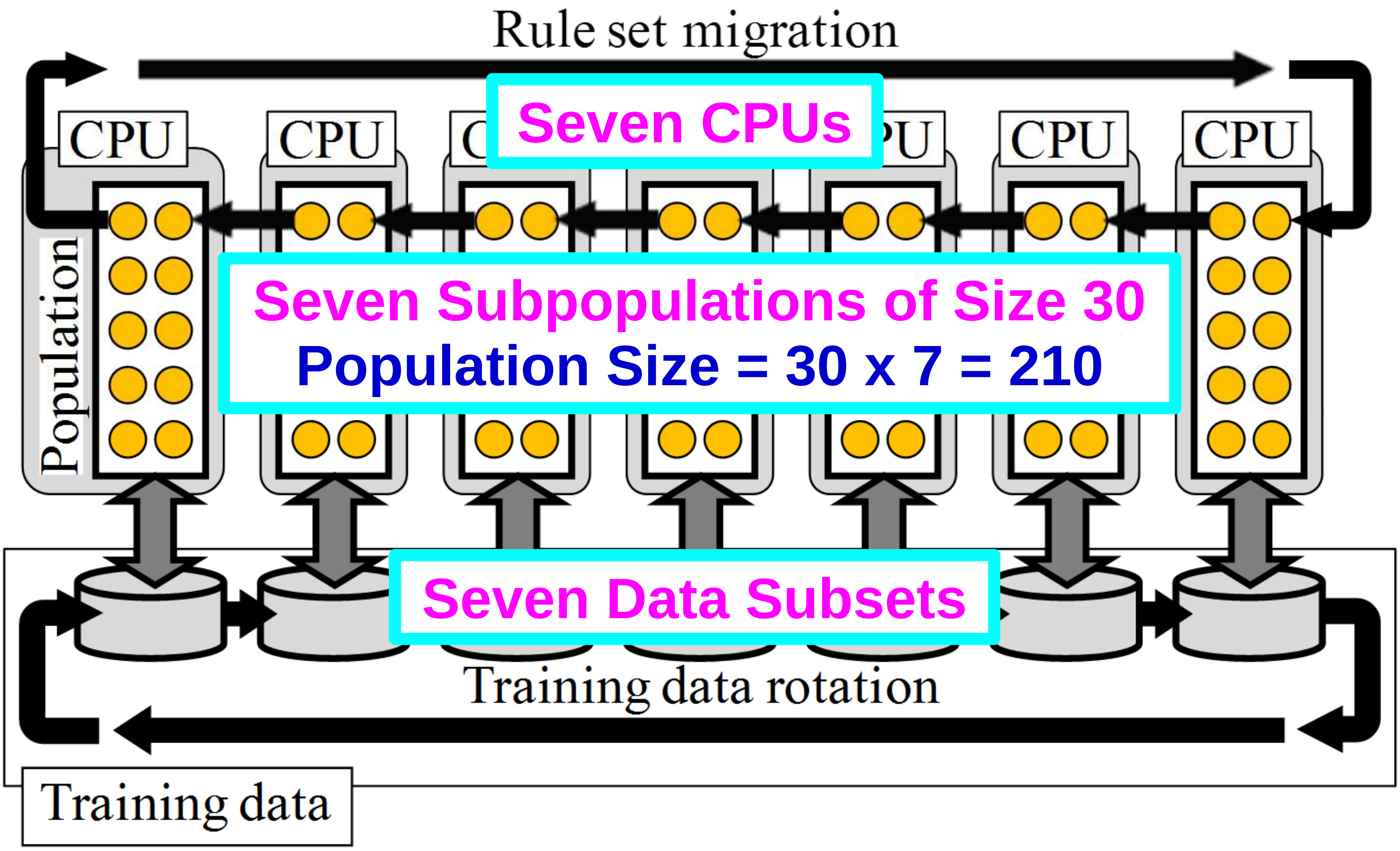
Contents of This Presentation

1. Basic Idea of Evolutionary Computation
2. Genetics-Based Machine Learning
3. Parallel Distributed Implementation
4. Computation Experiments
5. Conclusion

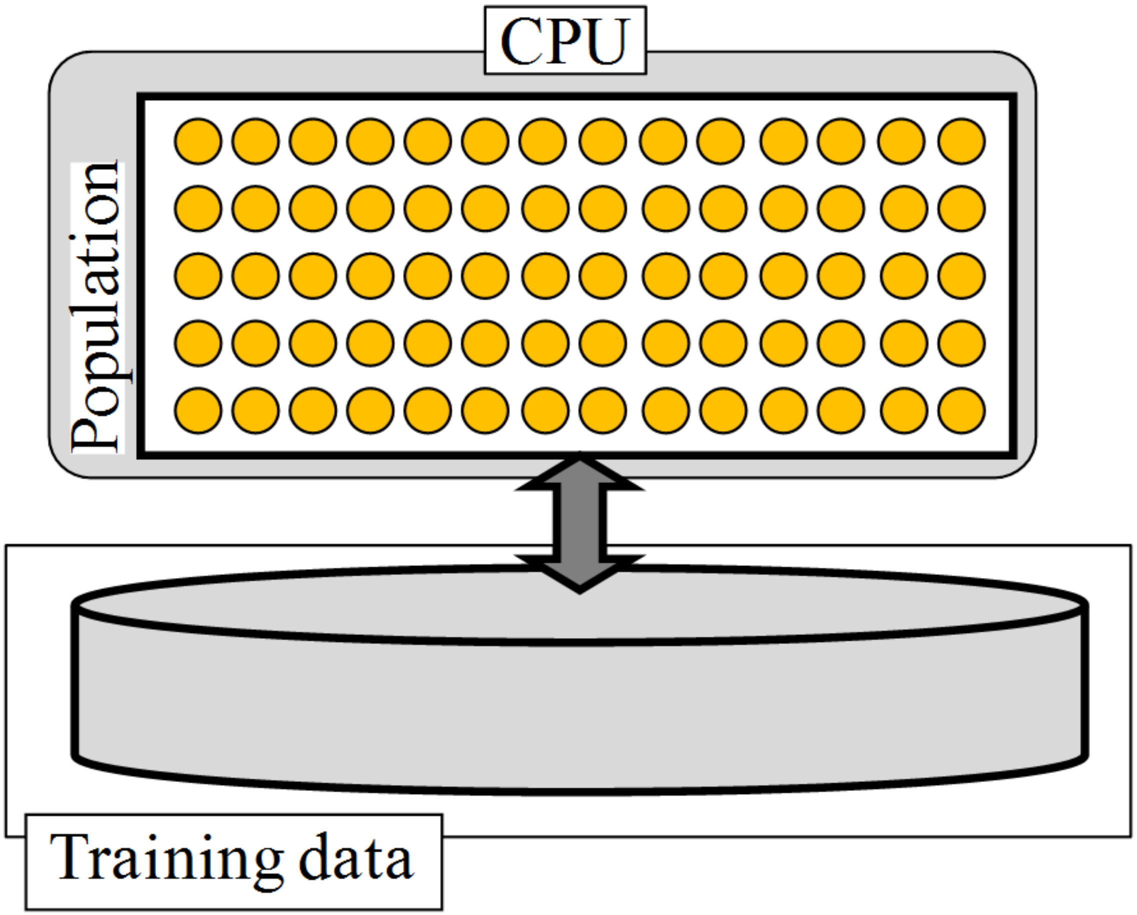
Our Model in Computational Experiments
with Seven Subpopulations and Seven Data Subsets



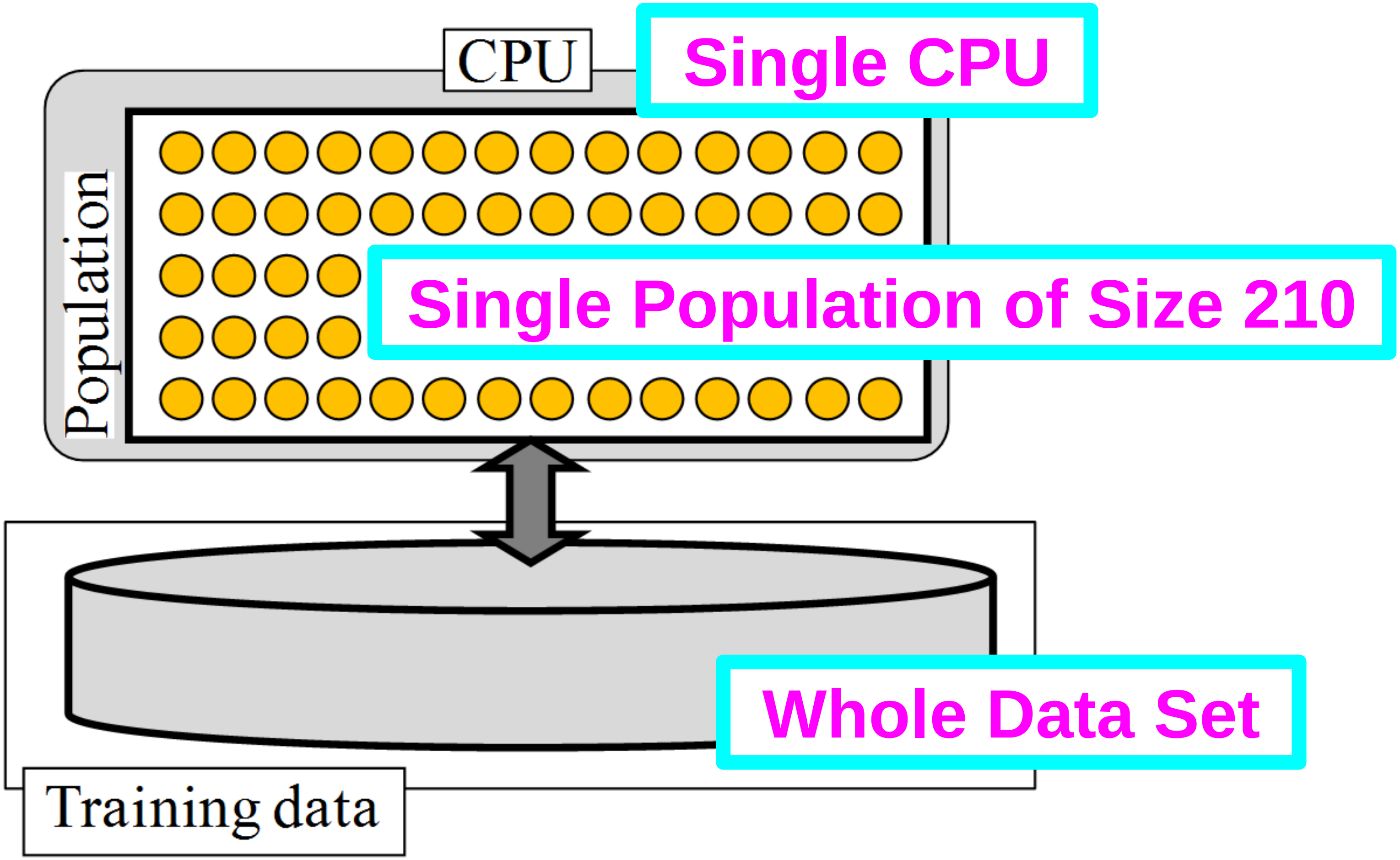
Our Model in Computational Experiments
with Seven Subpopulations and Seven Data Subsets



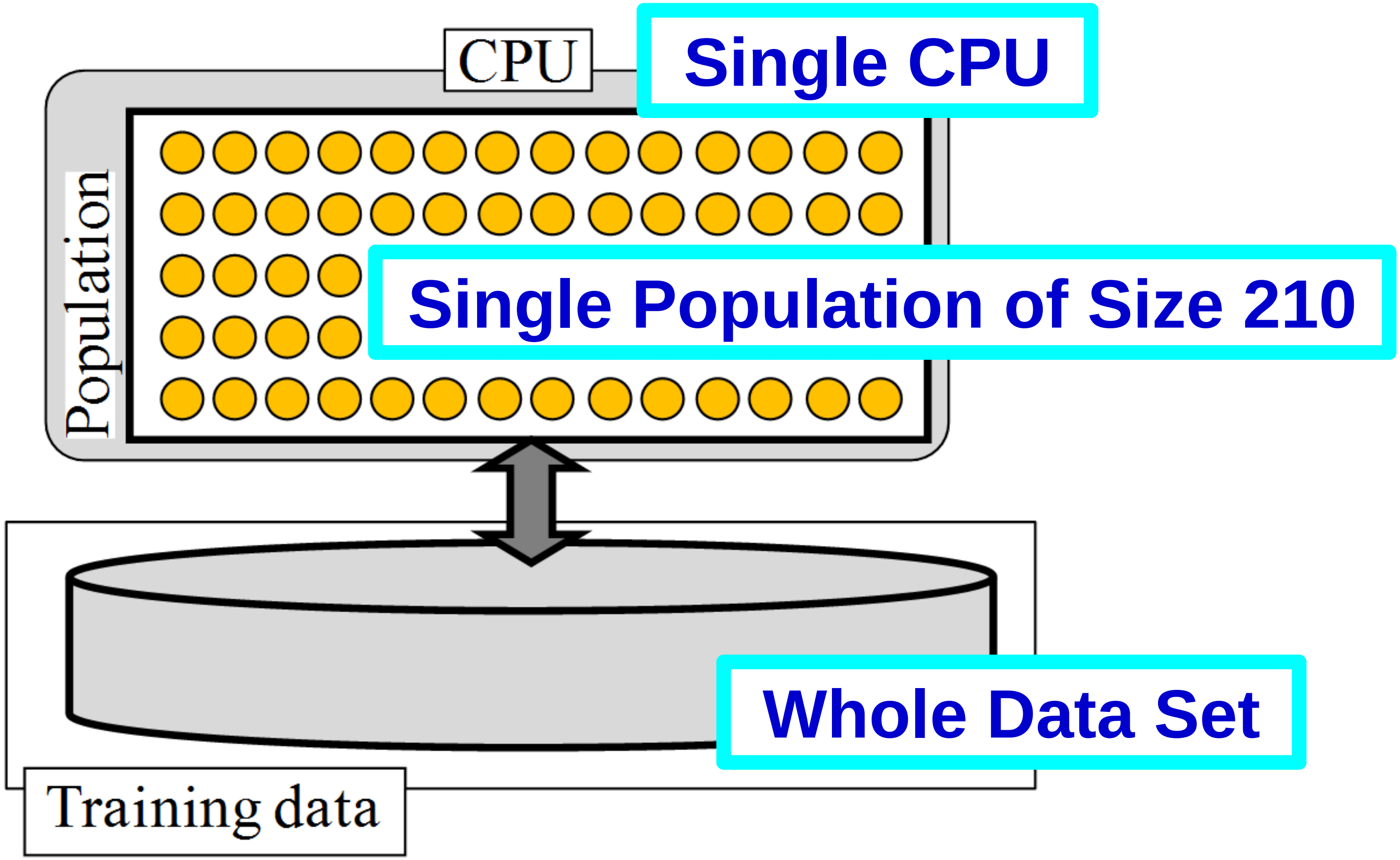
Standard Non-Parallel Non-Distributed Model
with a Single Population and a Single Data Set



Standard Non-Parallel Non-Distributed Model
with a Single Population and a Single Data Set



Standard Non-Parallel Non-Distributed Model
with a Single Population and a Single Data Set



Termination Conditions: 50,000 Generations

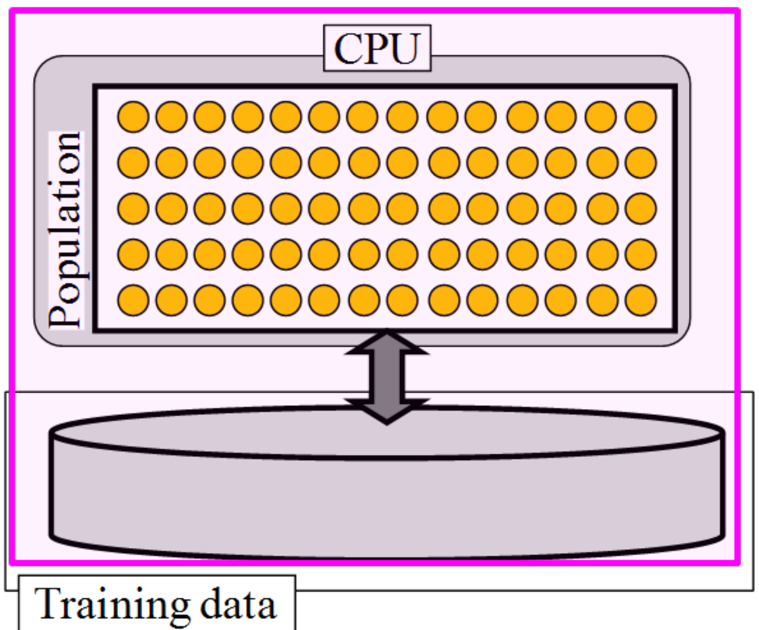
Computation Load: 210 x 50,000 = 10,500,000 Evaluations

(more than ten million evaluations)

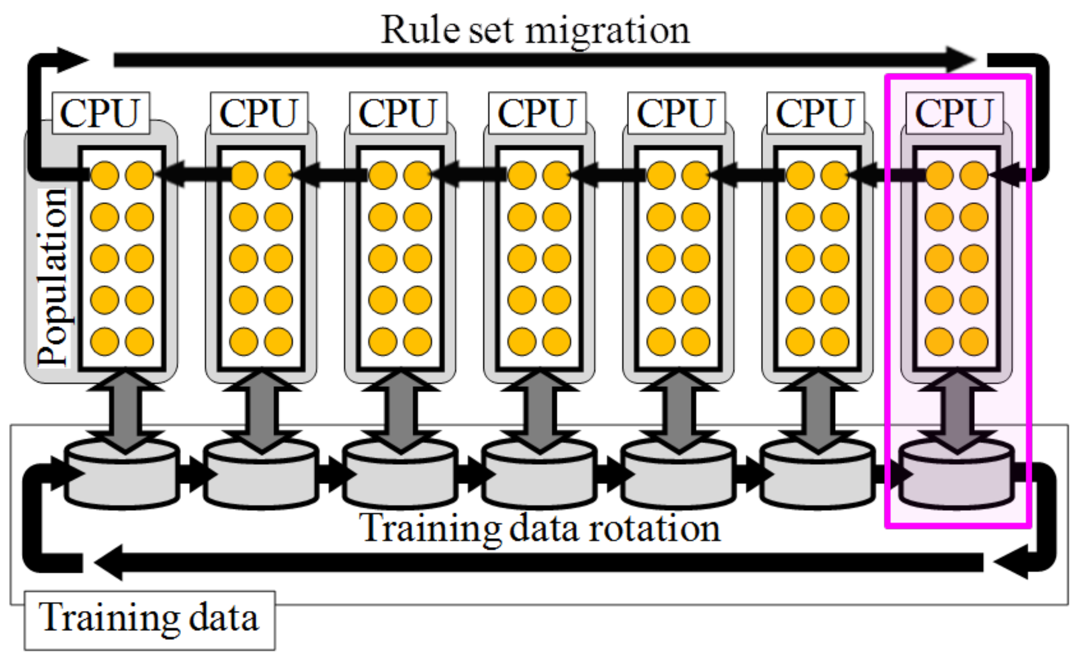
Comparison of Computation Load

Computation Load on a Single CPU per Generation

Standard Model:
Evaluation of 210 rule sets using all the training data



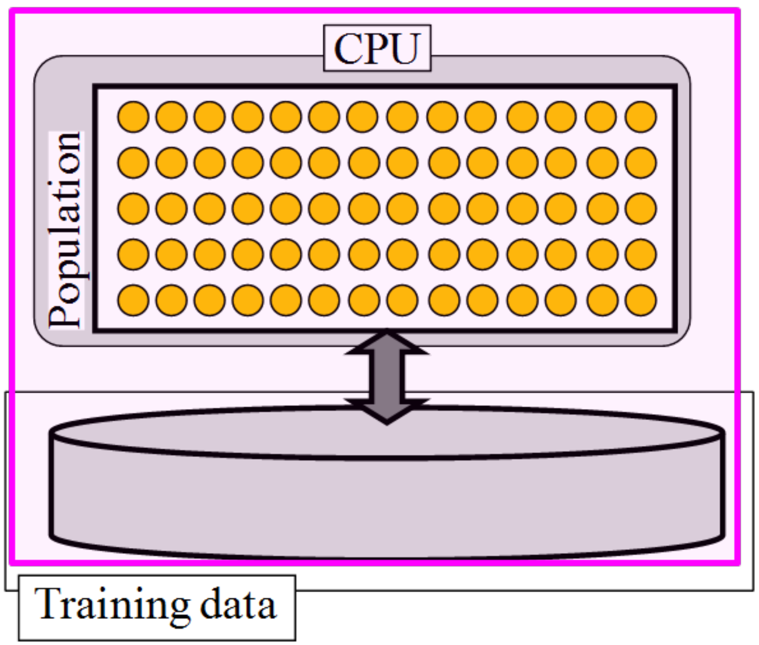
Parallel Distributed Model:
Evaluation of 30 rule sets using one of the seven data subsets.



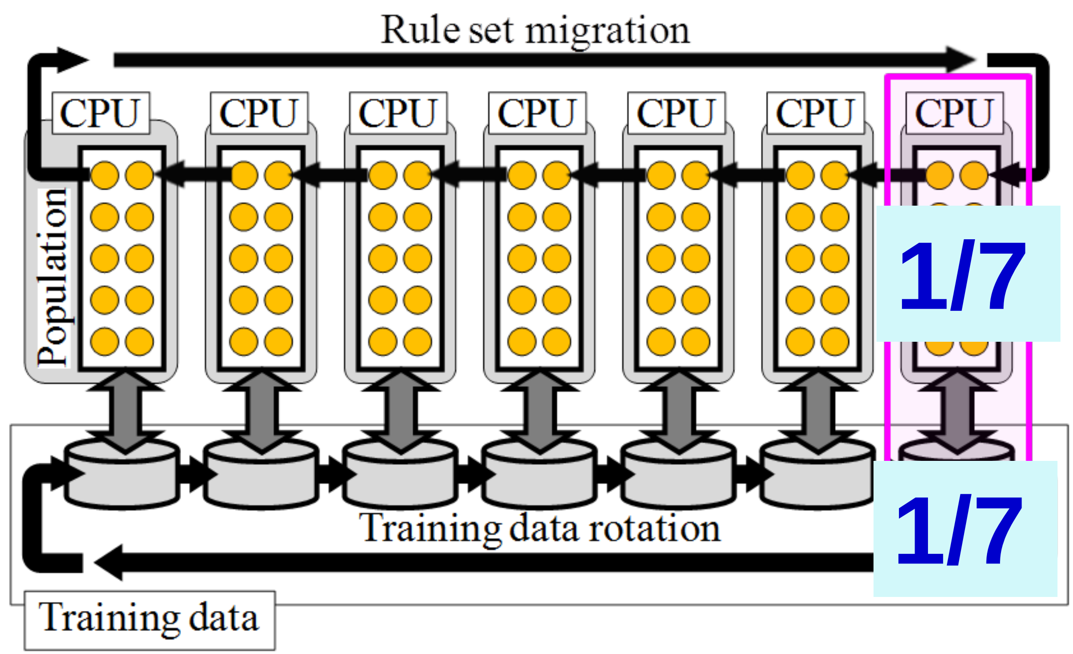
Comparison of Computation Load

Computation Load on a Single CPU per Generation

Standard Model:
Evaluation of 210 rule sets using all the training data



Parallel Distributed Model:
Evaluation of 30 rule sets using one of the seven data subsets.

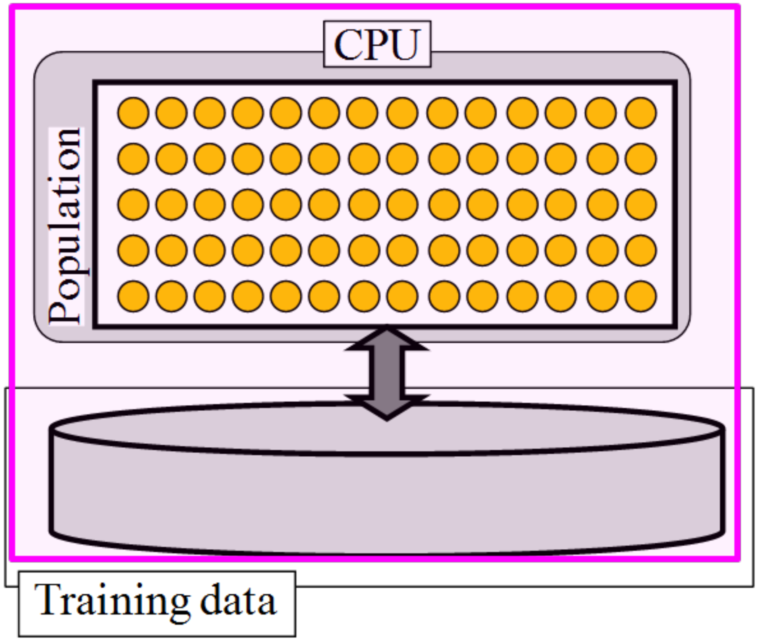


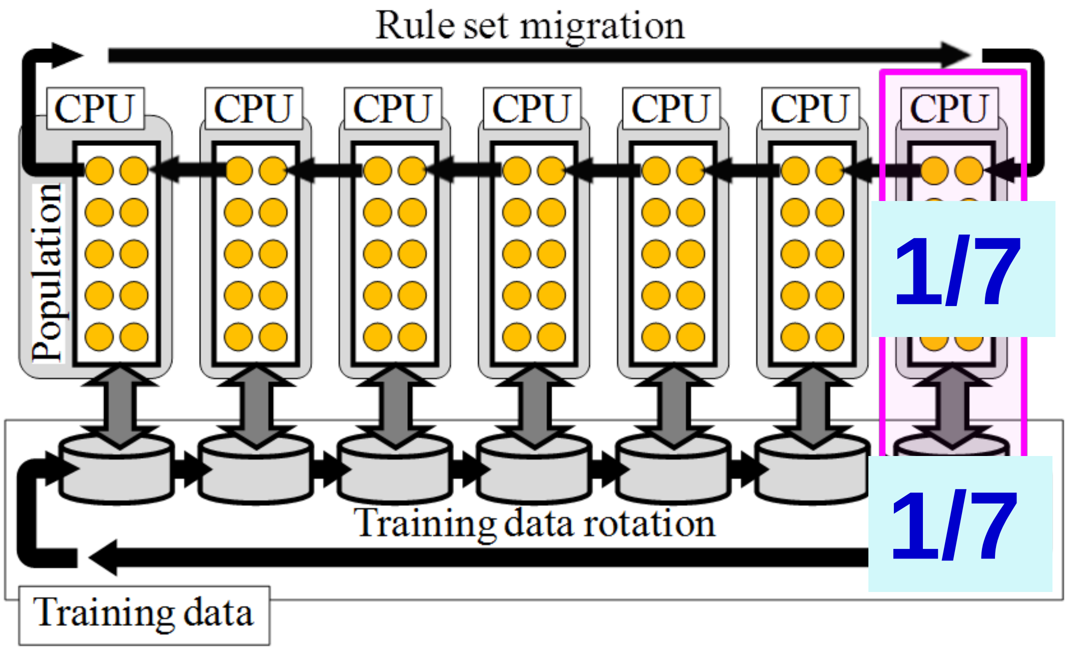
Comparison of Computation Load

Computation Load ==> 1/7 x 1/7 = 1/49 (about 2%)

Standard Model:
Evaluation of 210 rule sets using all the training data

Parallel Distributed Model:
Evaluation of 30 rule sets using one of the seven data subsets.





Data Sets in Computational Experiments

Nine Pattern Classification Problems

Name of Data Set	Number of Patterns	Number of Attributes	Number of Classes
Segment	2,310	19	7
Phoneme	5,404	5	2
Page-blocks	5,472	10	5
Texture	5,500	40	11
Satimage	6,435	36	6
Twonorm	7,400	20	2
Ring	7,400	20	2
PenBased	10,992	16	10
Magic	19,020	10	2

Computation Time for 50,000 Generations

Computation time was decreased to about 2%

Name of Data Set	Standard A minutes	Our Model B minutes	Percentage of B B/A (%)
Segment	203.66	4.69	2.30%
Phoneme	439.18	13.19	3.00%
Page-blocks	204.63	4.74	2.32%
Texture	766.61	15.72	2.05%
Satimage	658.89	15.38	2.33%
Twonorm	856.58	7.84	0.92%
Ring	1015.04	22.52	2.22%
PenBased	1520.54	35.56	2.34%
Magic	771.05	22.58	2.93%

Computation Time for 50,000 Generations

Computation time was decreased to about 2%

Name of Data Set	Standard A minutes	Our Model B minutes	Percentage of B B/A (%)
Se			
Ph			
Pag			
Te			
Sa			
Twonorm	856.58	7.84	0.92%
Ring	1015.04	22.52	2.22%
PenBased	1520.54	35.56	2.34%
Magic	771.05	22.58	2.93%

Why ?

Computation Time for 50,000 Generations

Computation time was decreased to about 2%

Name of Data Set	Standard A minutes	Our Model B minutes	Percentage of B B/A (%)
Se			
Ph			
Pag			
Te			
Sa			
Twonorm	856.58	7.84	0.92%
Ring	1015.04	22.52	2.22%
PenBased	1520.54	35.56	2.34%
Magic	771.05	22.58	2.93%

Why ?

Because the population and the training data were divided into seven subsets.

1/7 x 1/7 = 1/49 (about 2%)

Test Data Error Rates (Results of 3x10CV)

Test data accuracy was improved for six data sets

Name of Data Set	Standard (A %)	Our Model (B %)	Improvement from A: (A - B)%
Segment	5.99	5.90	0.09
Phoneme	15.43	15.96	- 0.53
Page-blocks	3.81	3.62	0.19
Texture	4.64	4.77	- 0.13
Satimage	15.54	12.96	2.58
Twonorm	7.36	3.39	3.97
Ring	6.73	5.25	1.48
PenBased	3.07	3.30	- 0.23
Magic	15.42	14.89	0.53

Test Data Error Rates (Results of 3x10CV)

Test data accuracy was improved for six data sets

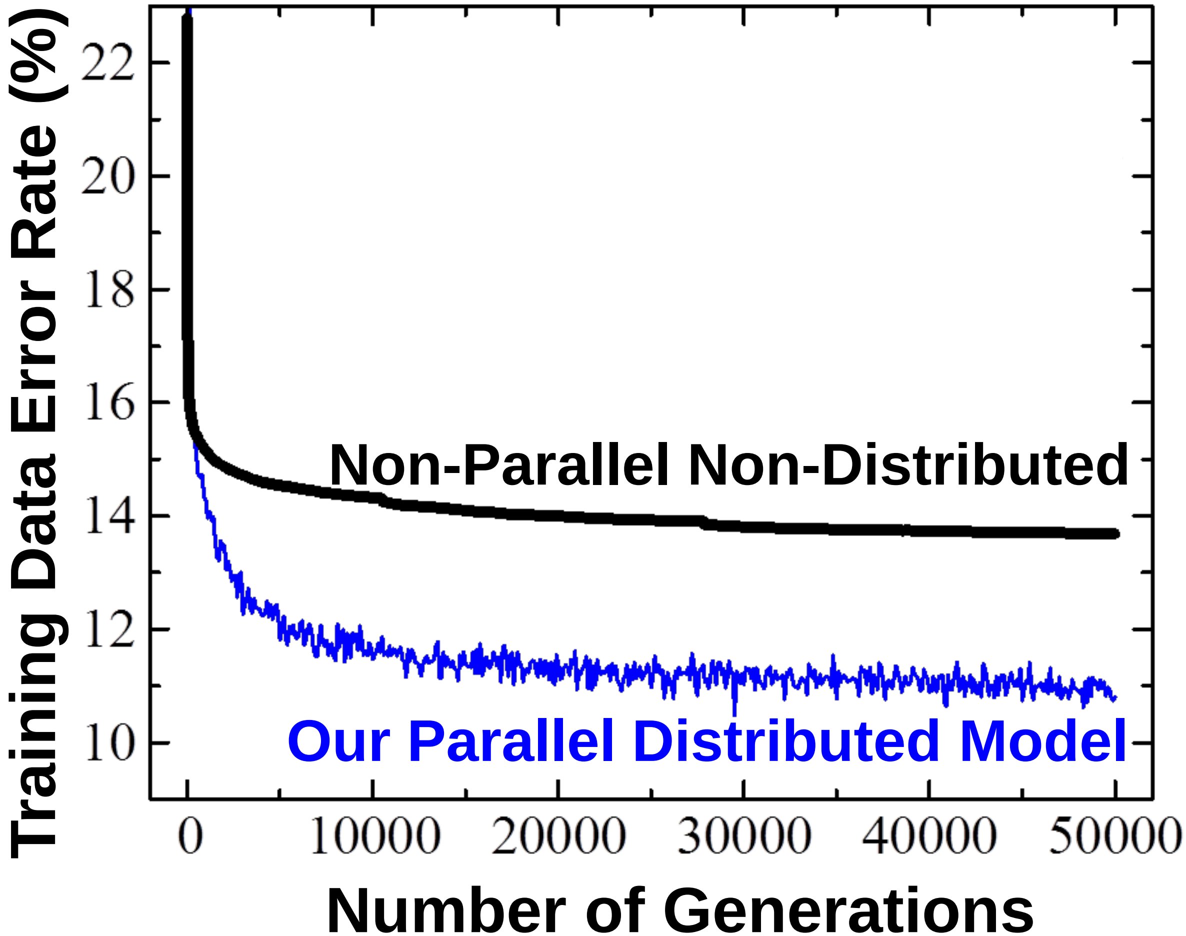
Name of Data Set	Standard (A %)	Our Model (B %)	Improvement from A: (A - B)%
Segment	5.99	5.90	0.09
Phoneme	15.43	15.96	- 0.53
Page-blocks	3.81	3.62	0.19
Texture	4.64	4.77	- 0.13
Satimage	15.54	12.96	2.58
Twonorm	7.36	3.39	3.97
Ring	6.73	5.25	1.48
PenBased	3.07	3.30	- 0.23
Magic	15.42	14.89	0.53

Why ?

Q. Why did our model improve the test data accuracy ?

A. Because our model improved the search ability.

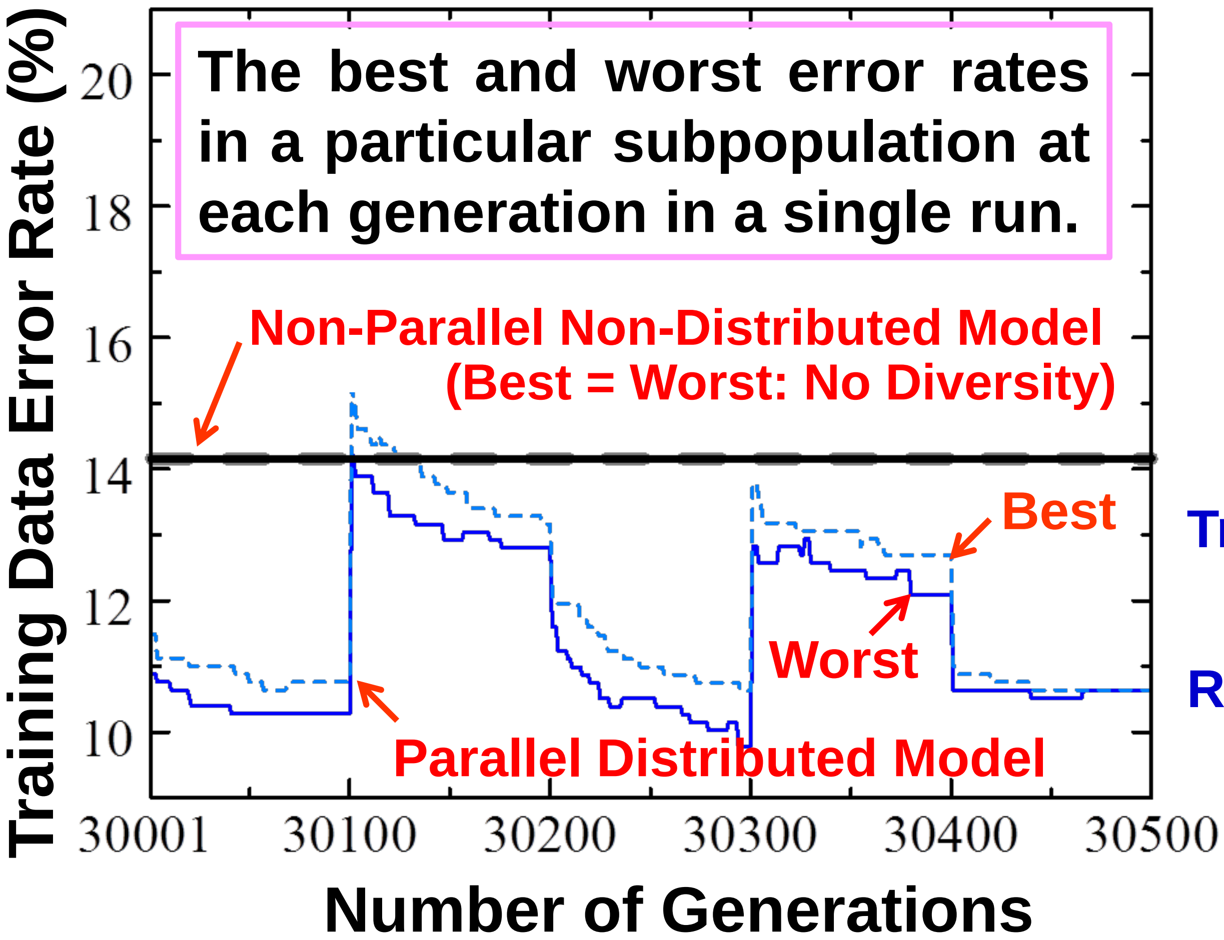
Data Set	Standard	Our Model	Improvement
Satimage	15.54%	12.96%	2.58%



Q. Why did our model improve the search ability ?

A. Because our model maintained the diversity.

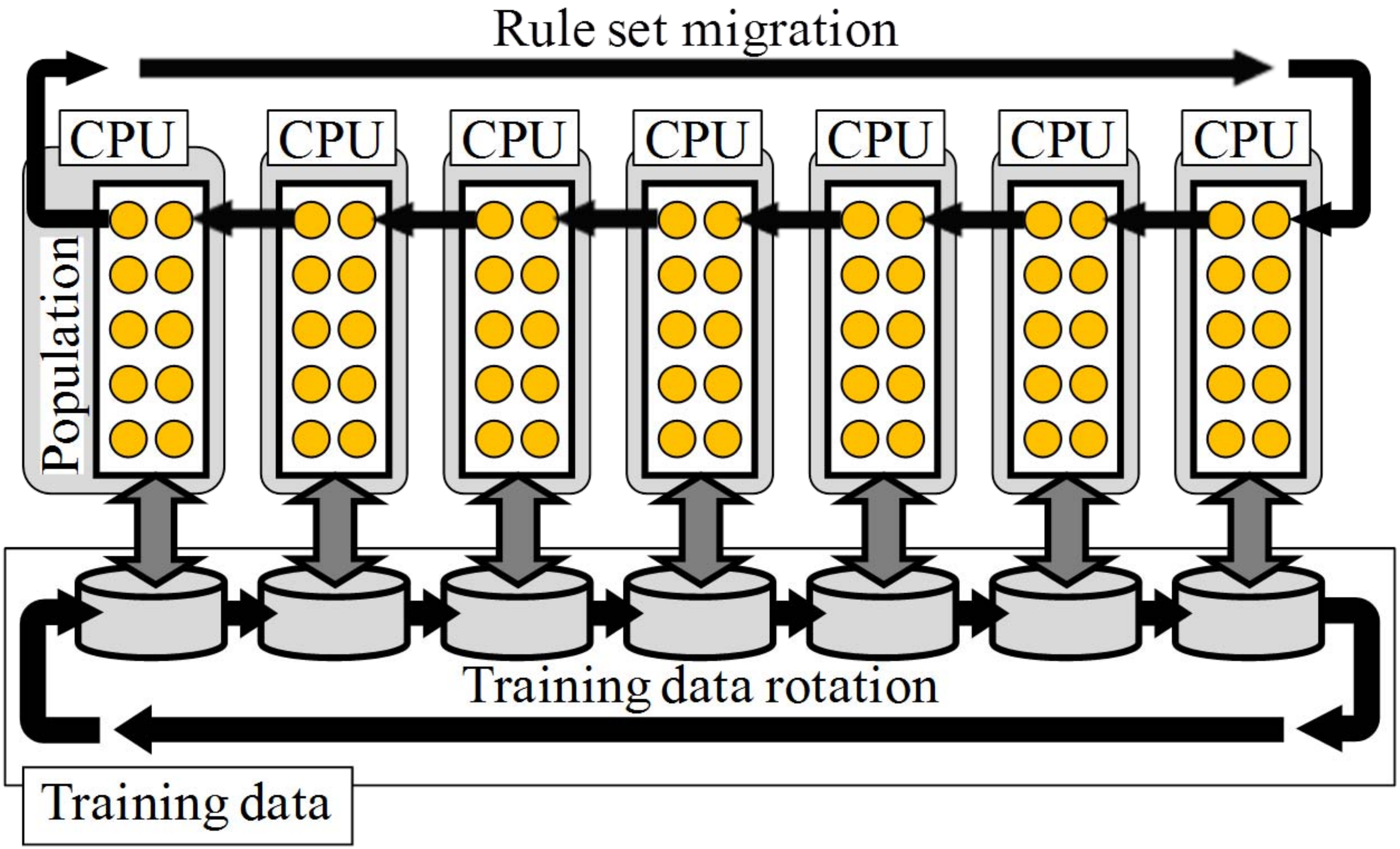
Data Set	Standard	Our Model	Improvement
Satimage	15.54%	12.96%	2.58%



Training Data Rotation:
Every 100 Generations

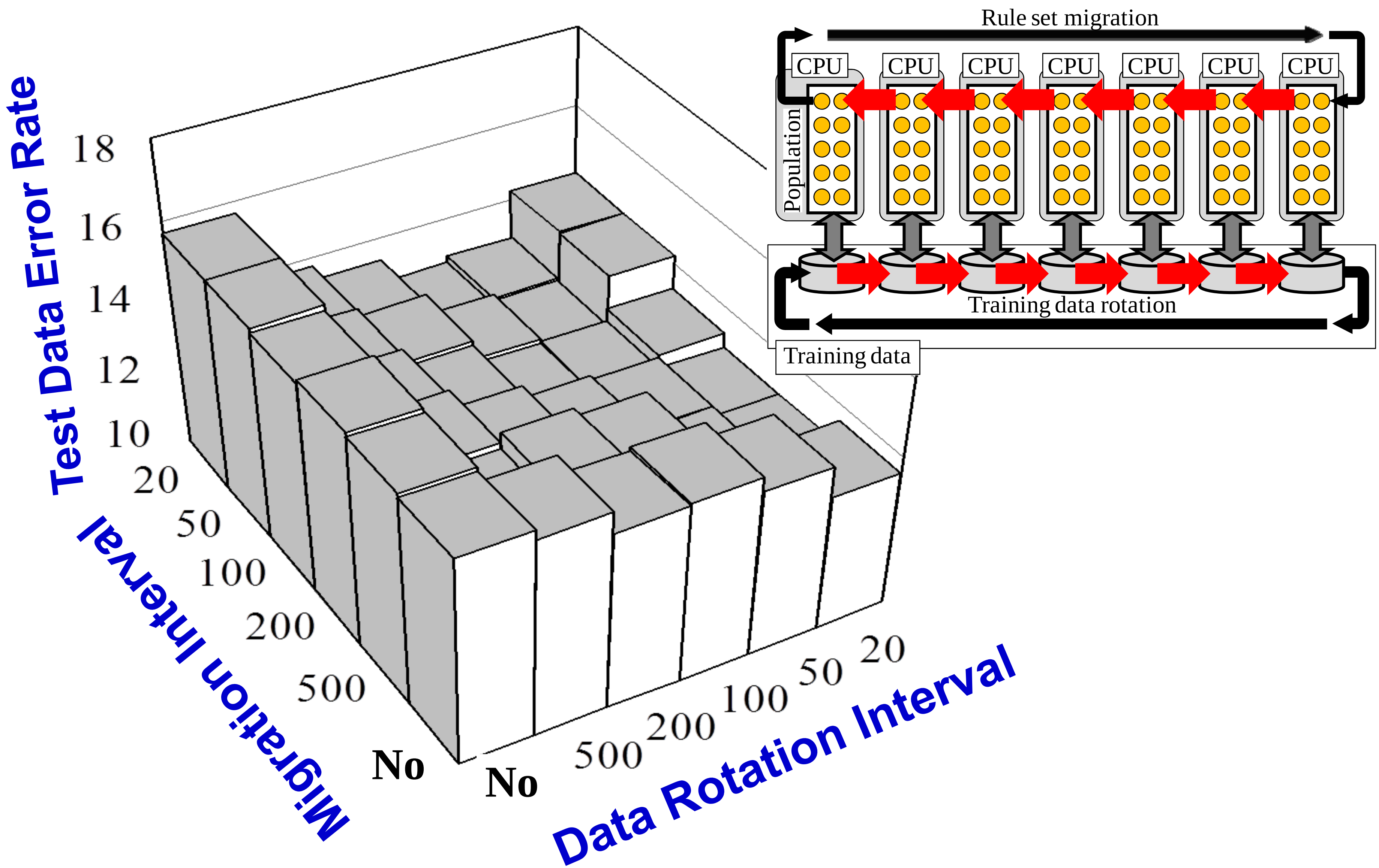
Rule Set Migration:
Every 100 Generations

Effects of Rotation and Migration Intervals

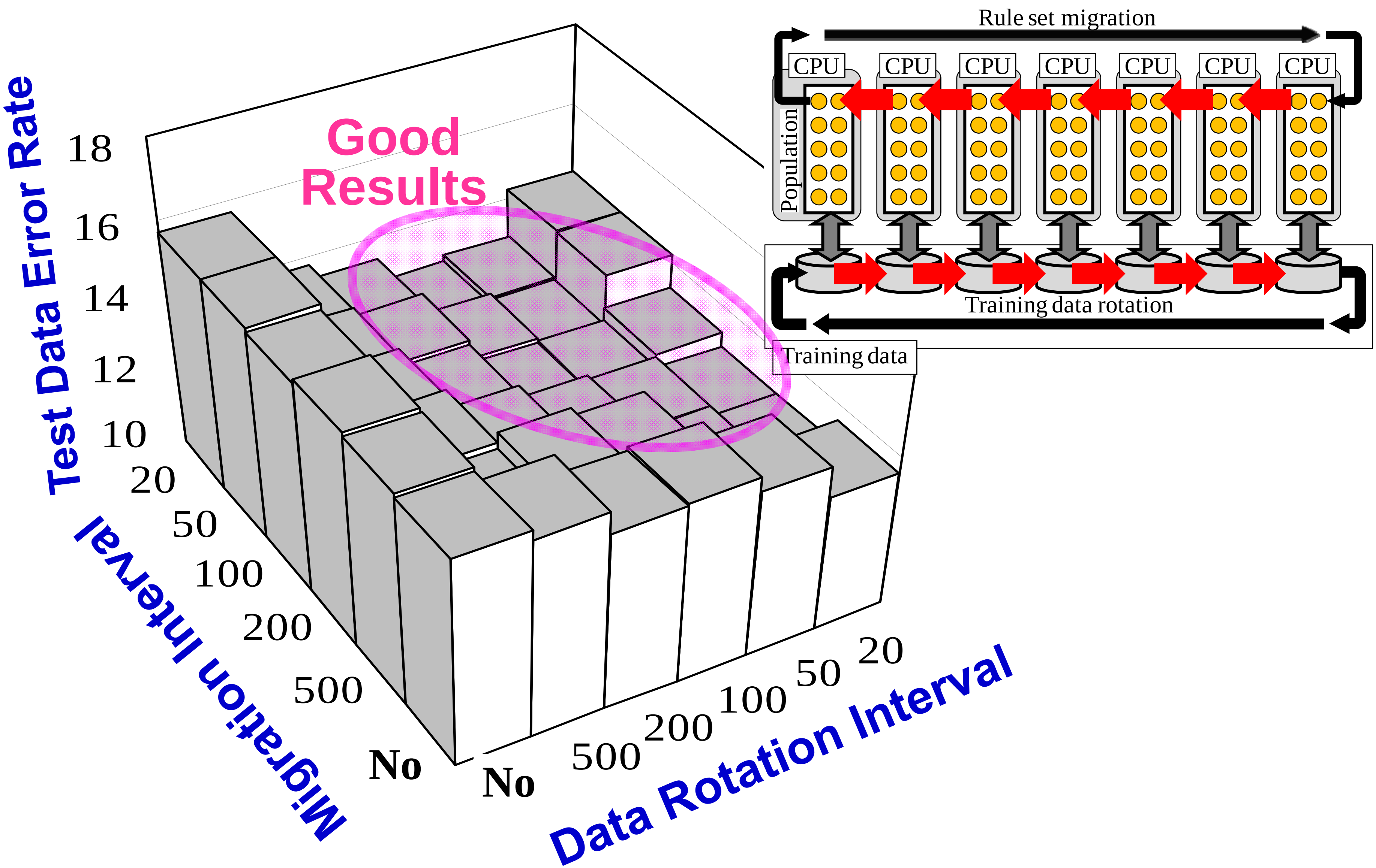


Training Data Rotation: Every 100 Generations
Rule Set Migration: Every 100 Generations

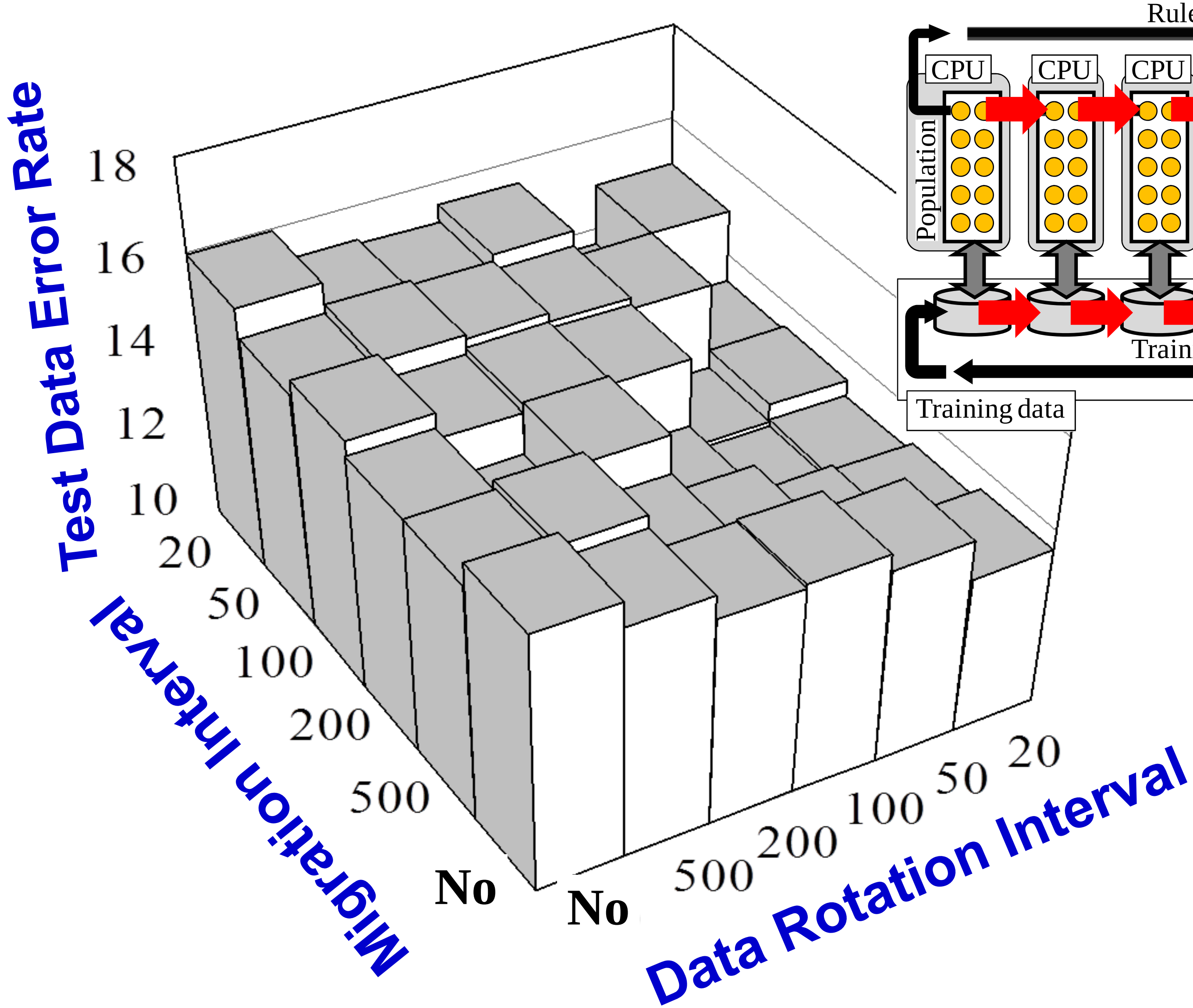
Effects of Rotation and Migration Intervals
(Rotations in the opposite directions)

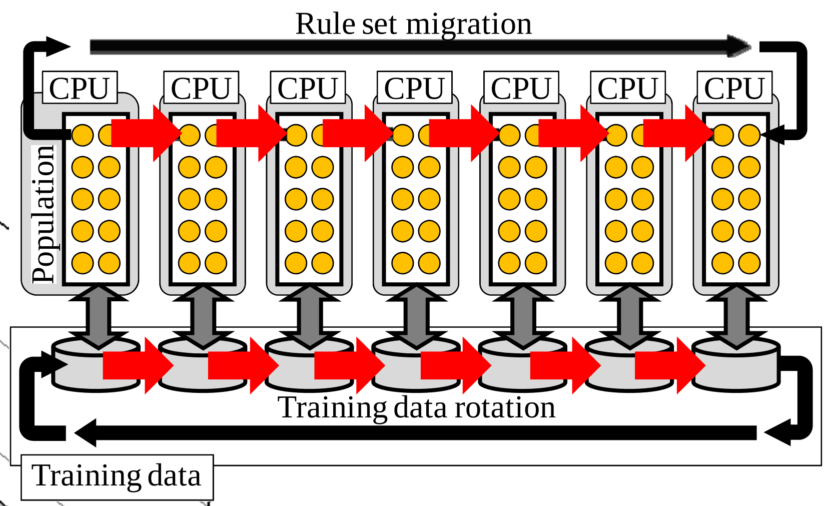


Effects of Rotation and Migration Intervals
(Rotations in the opposite directions)

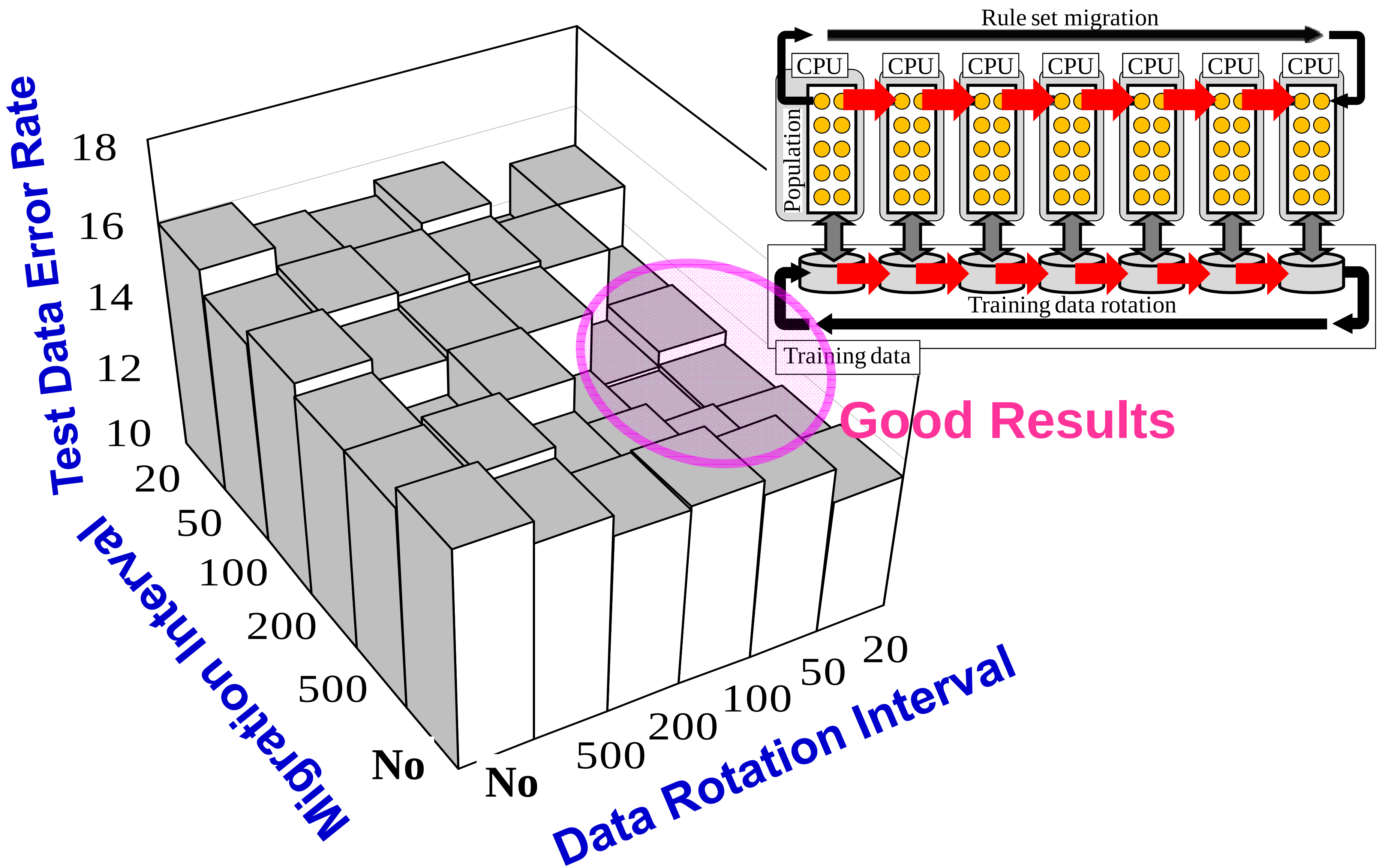


Effects of Rotation and Migration Intervals
(Rotations in the same direction)

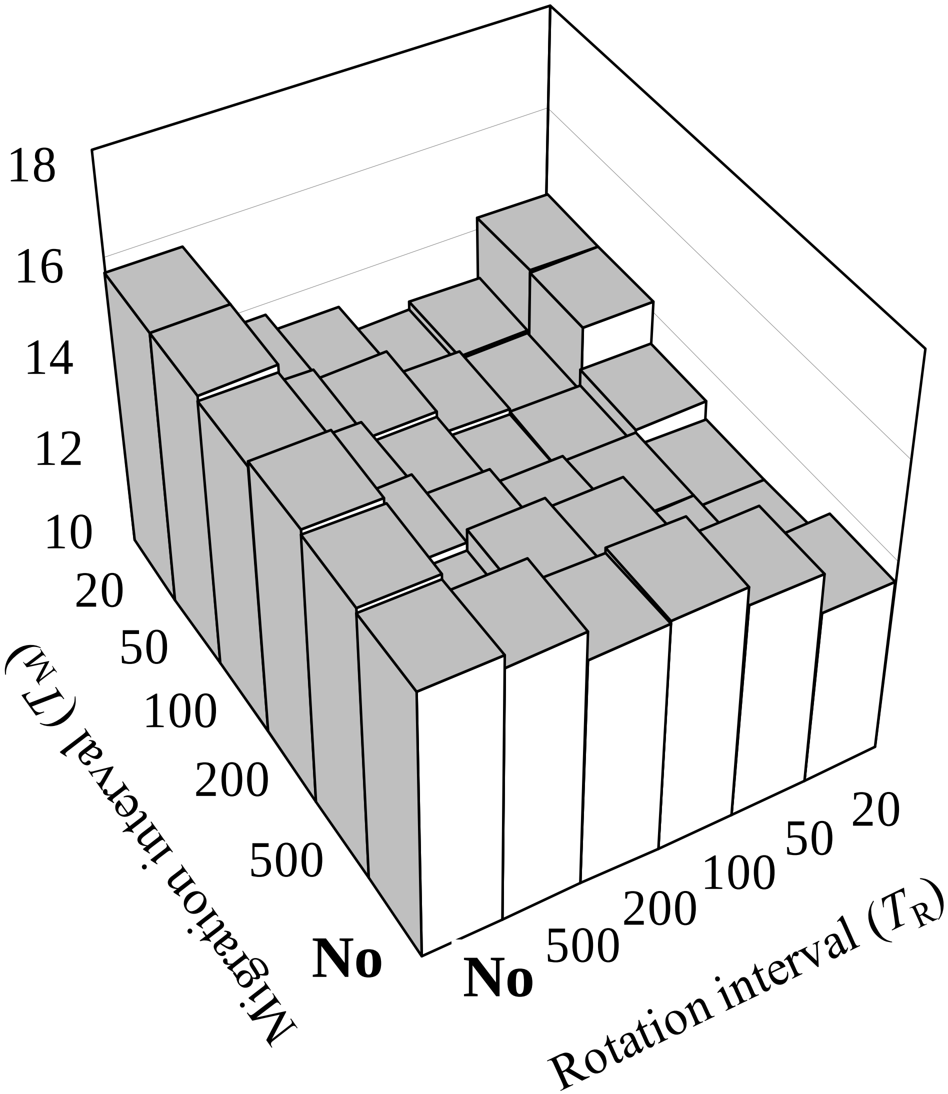




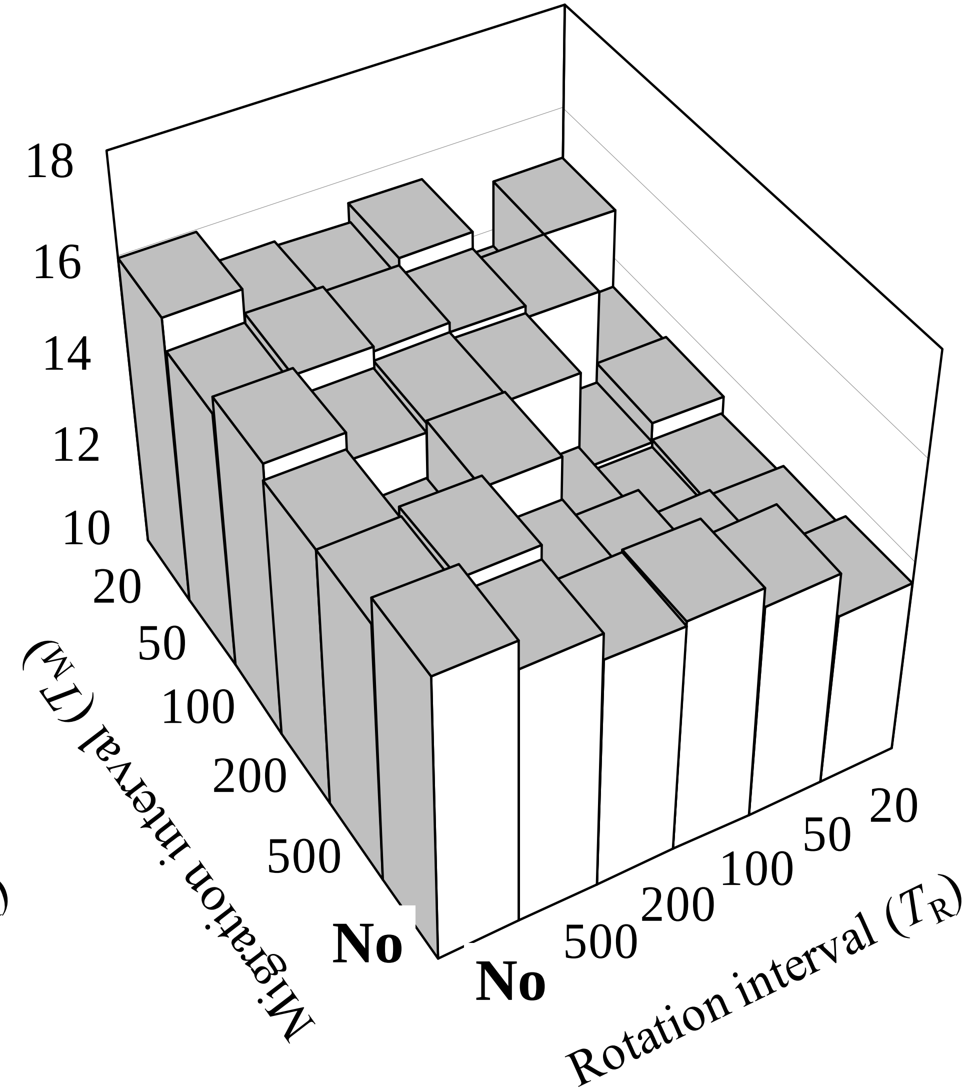
Effects of Rotation and Migration Intervals
(Rotations in the same direction)



Effects of Rotation and Migration Intervals

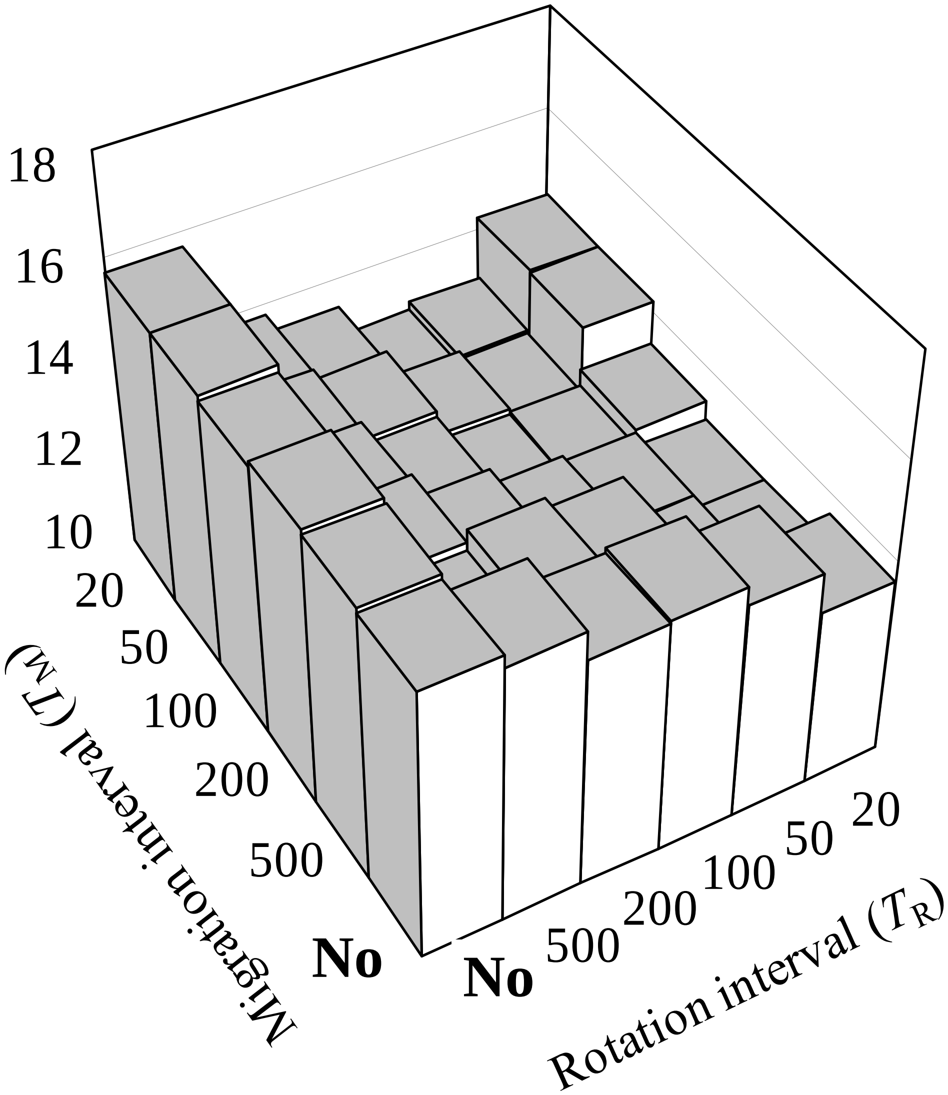


Opposite Directions



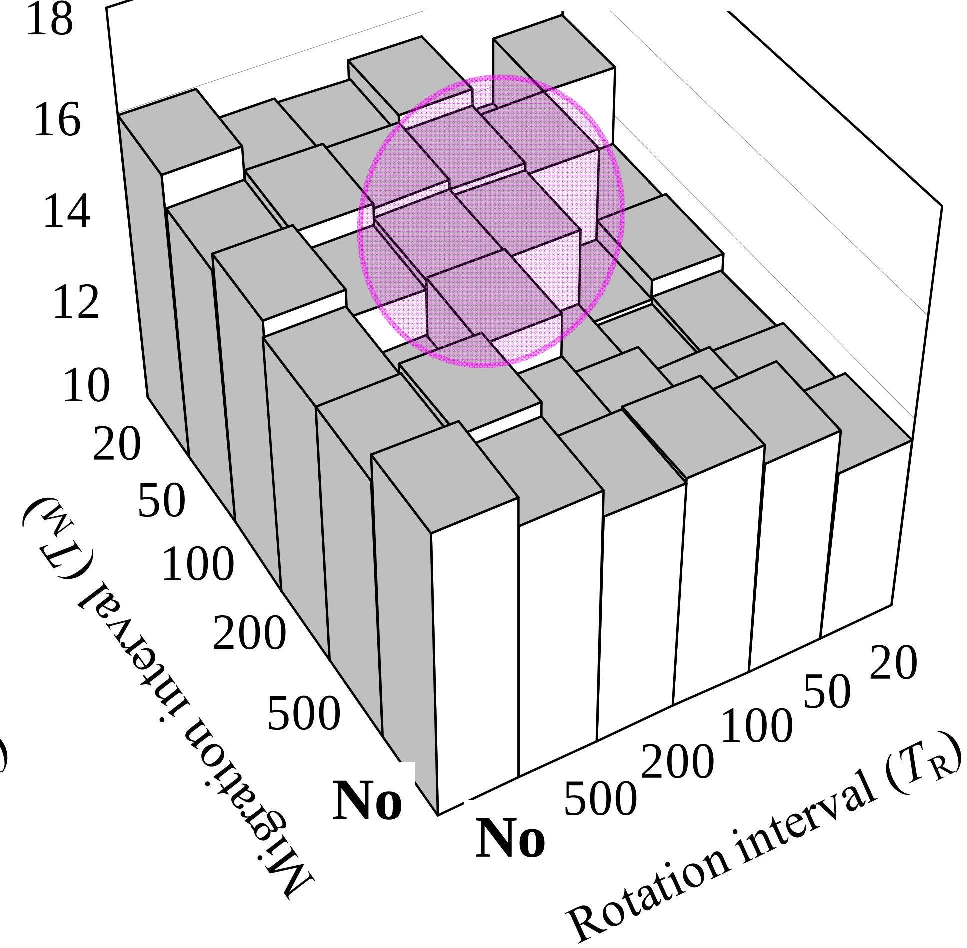
Same Direction

Effects of Rotation and Migration Intervals



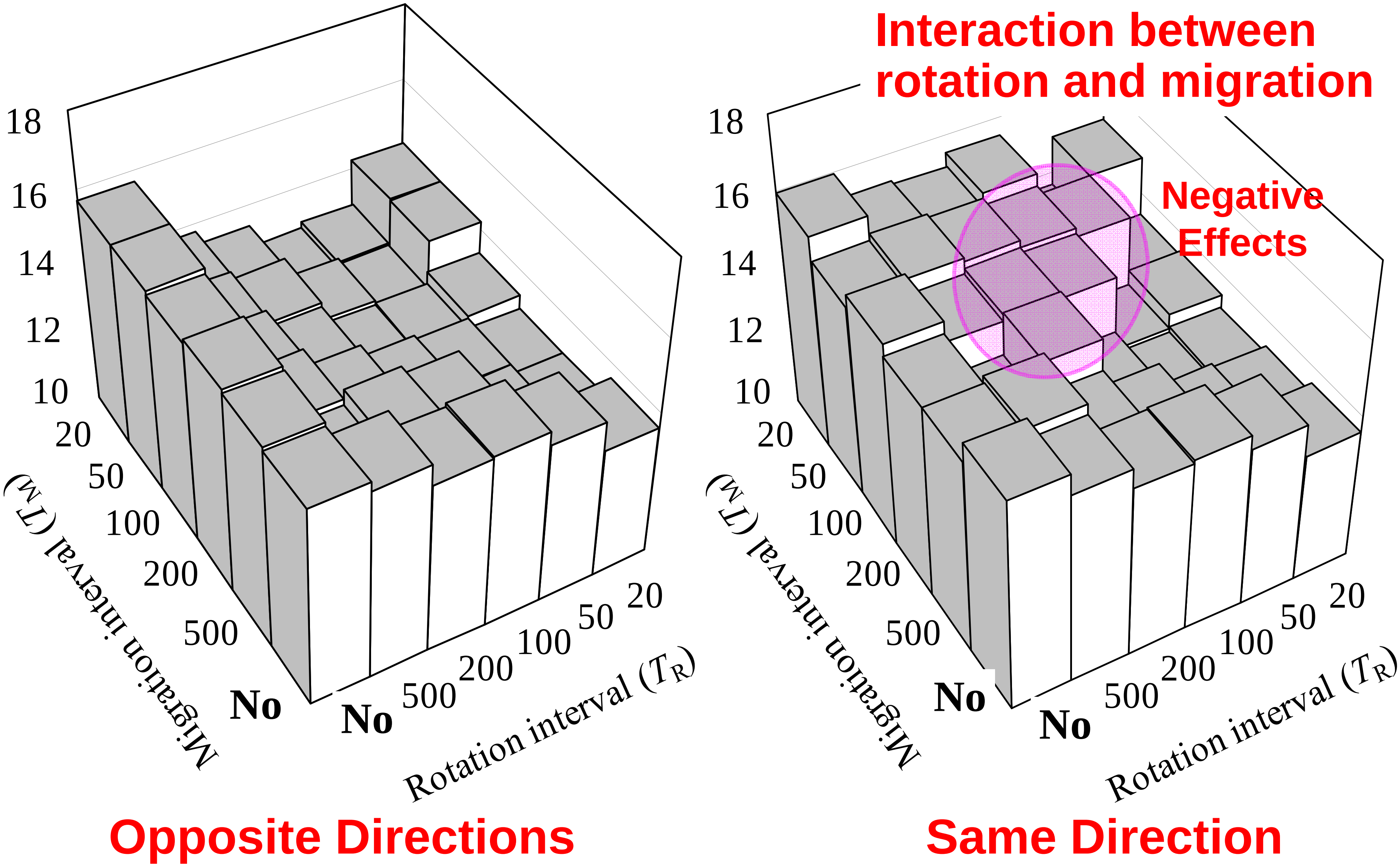
Opposite Directions

Interaction between rotation and migration



Same Direction

Effects of Rotation and Migration Intervals

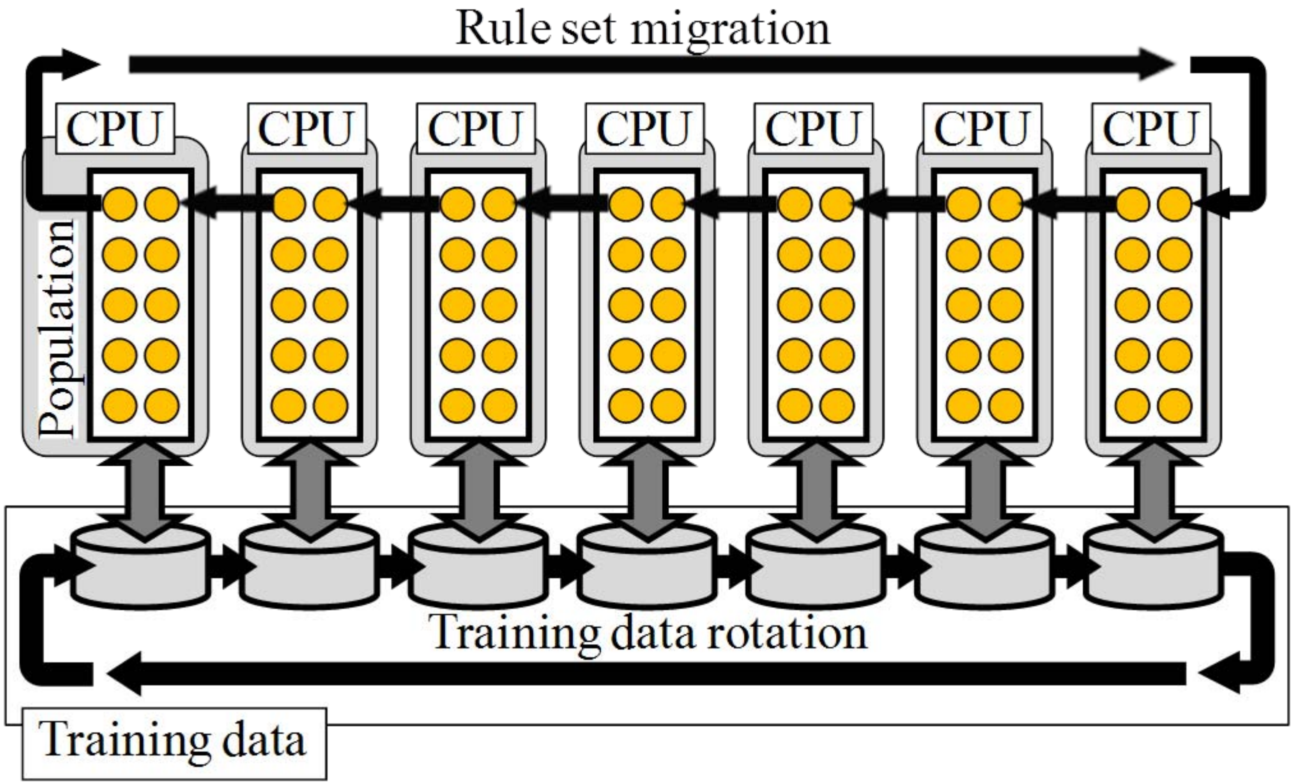


Contents of This Presentation

1. Basic Idea of Evolutionary Computation
2. Genetics-Based Machine Learning
3. Parallel Distributed Implementation
4. Computation Experiments
5. Conclusion

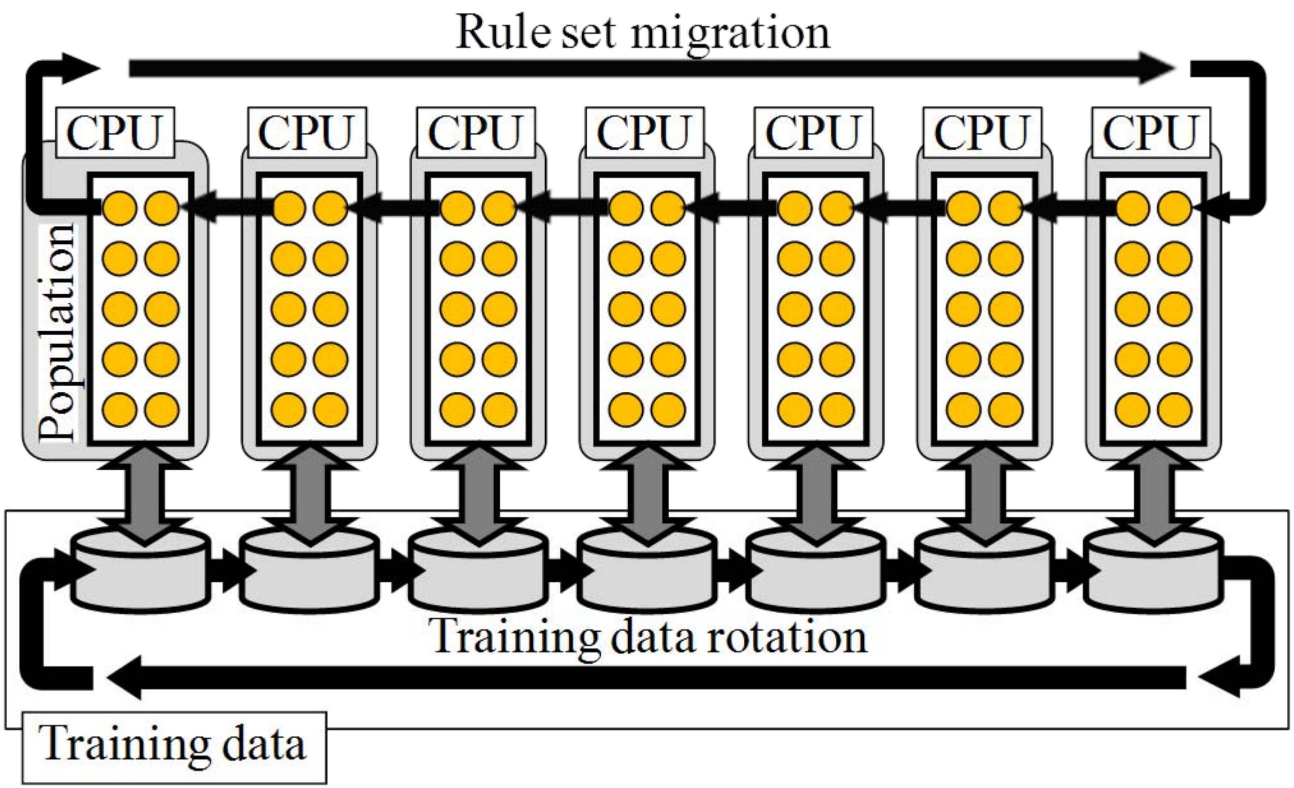
Conclusion

1. We explained our parallel distributed model.



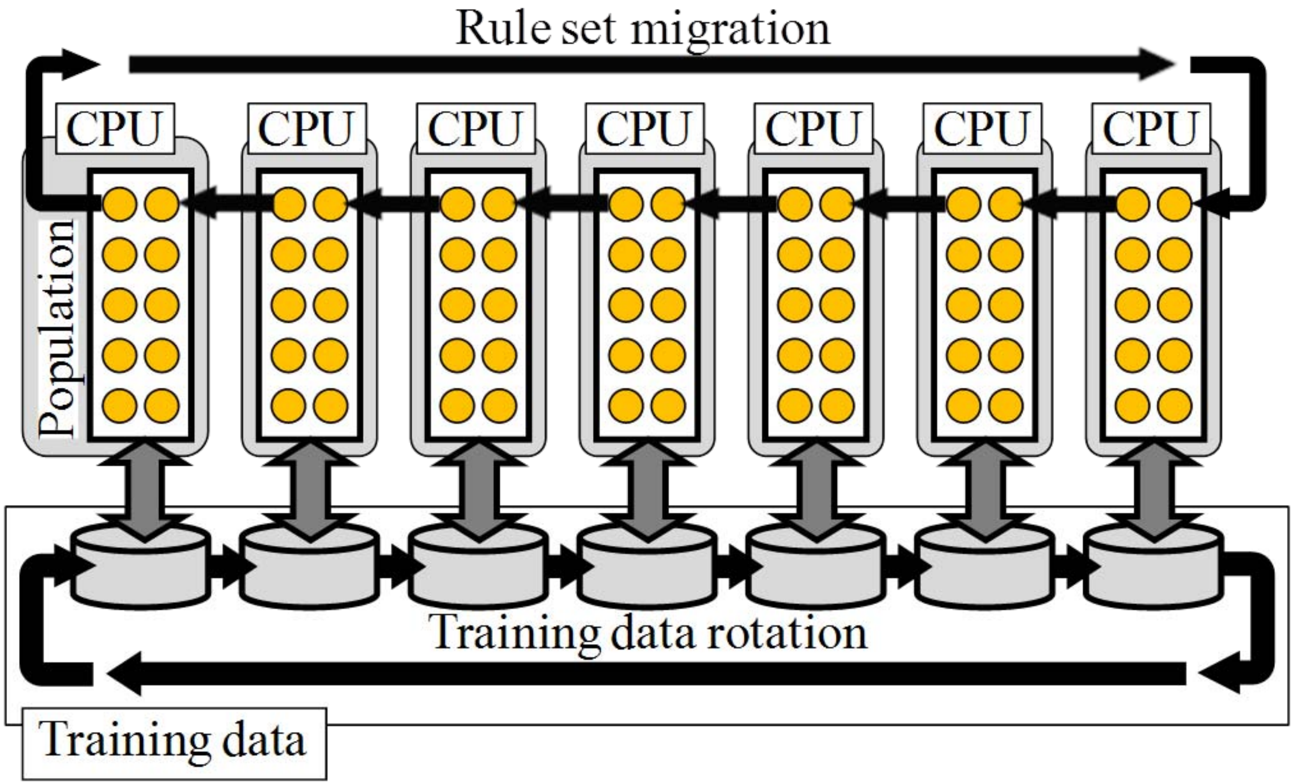
Conclusion

1. We explained our parallel distributed model.
2. It was shown that the computation time was decreased to 2%.



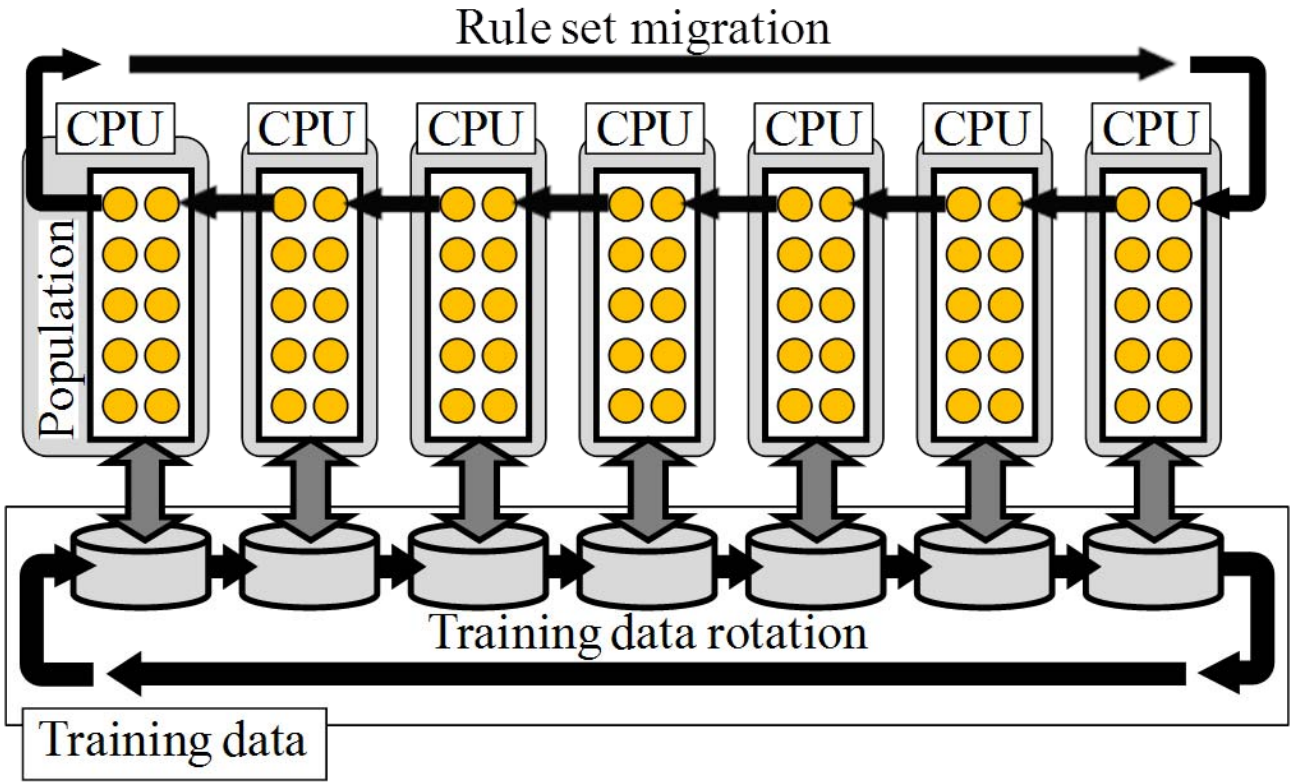
Conclusion

1. We explained our parallel distributed model.
2. It was shown that the computation time was decreased to 2%.
3. It was shown that the test data accuracy was improved.



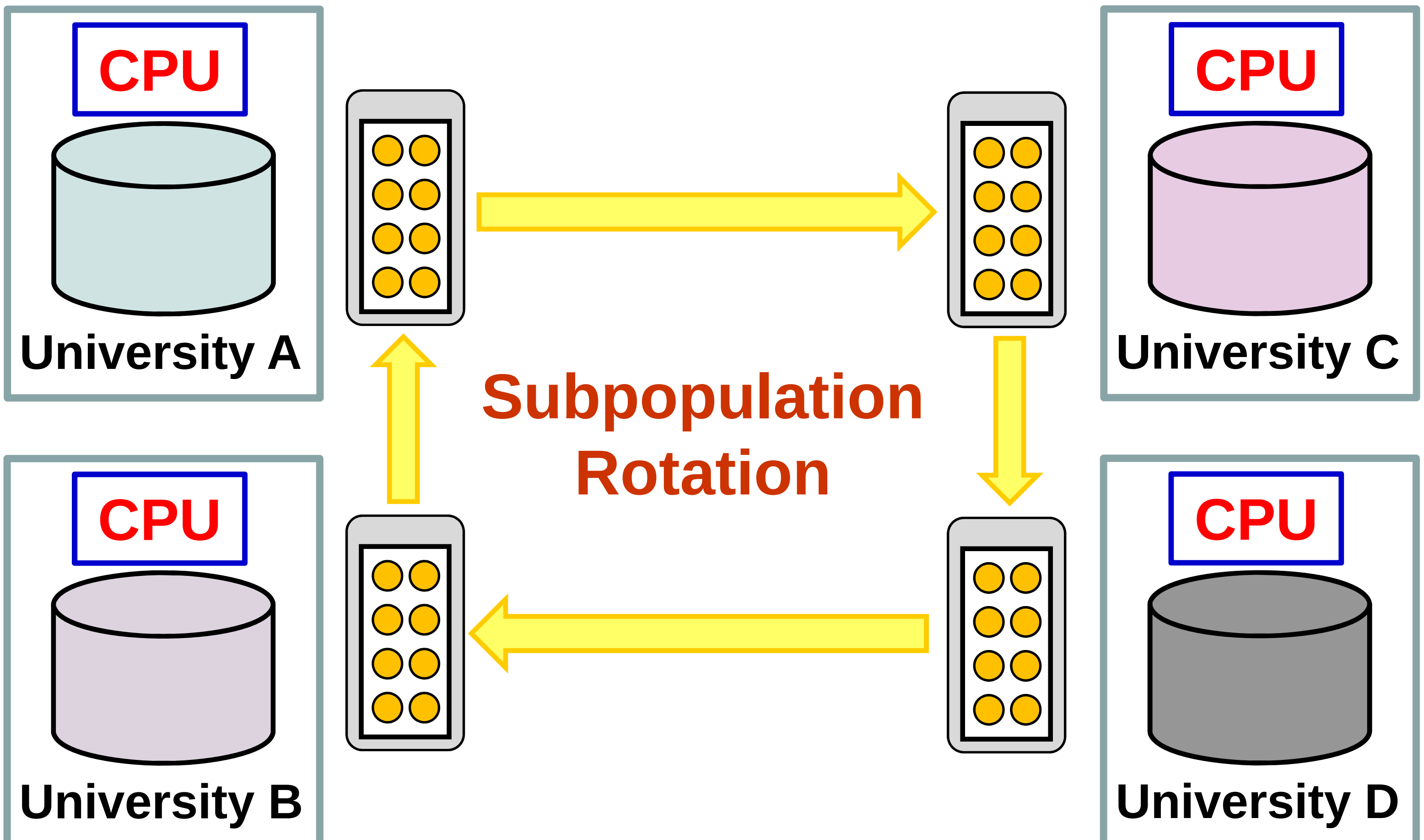
Conclusion

1. We explained our parallel distributed model.
2. It was shown that the computation time was decreased to 2%.
3. It was shown that the test data accuracy was improved.
4. We explained negative effects of the interaction between the training data rotation and the rule set migration.



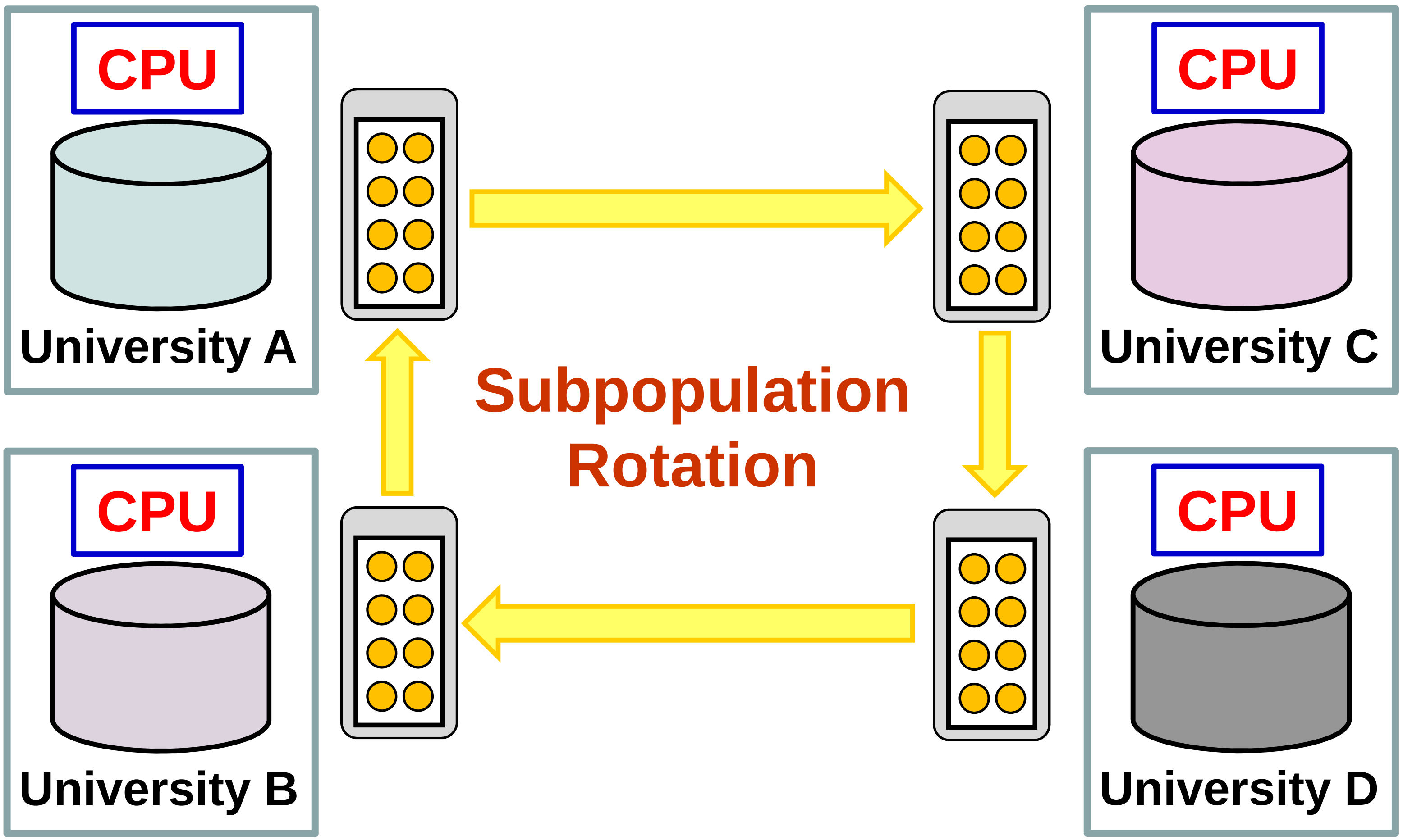
Conclusion

5. A little bit different model may be also possible for learning from locally located data bases.



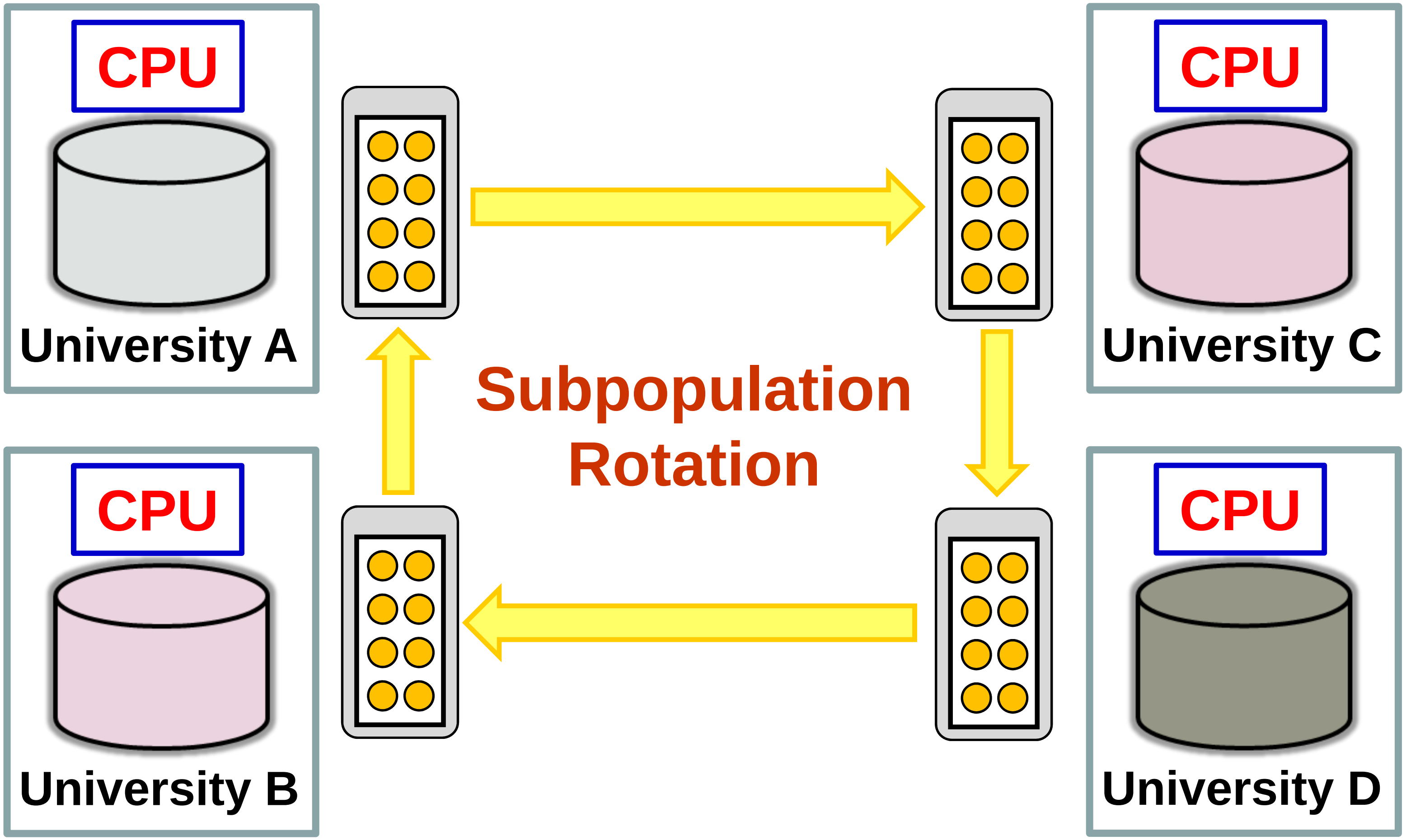
Conclusion

6. This model can be also used for learning from changing data bases.



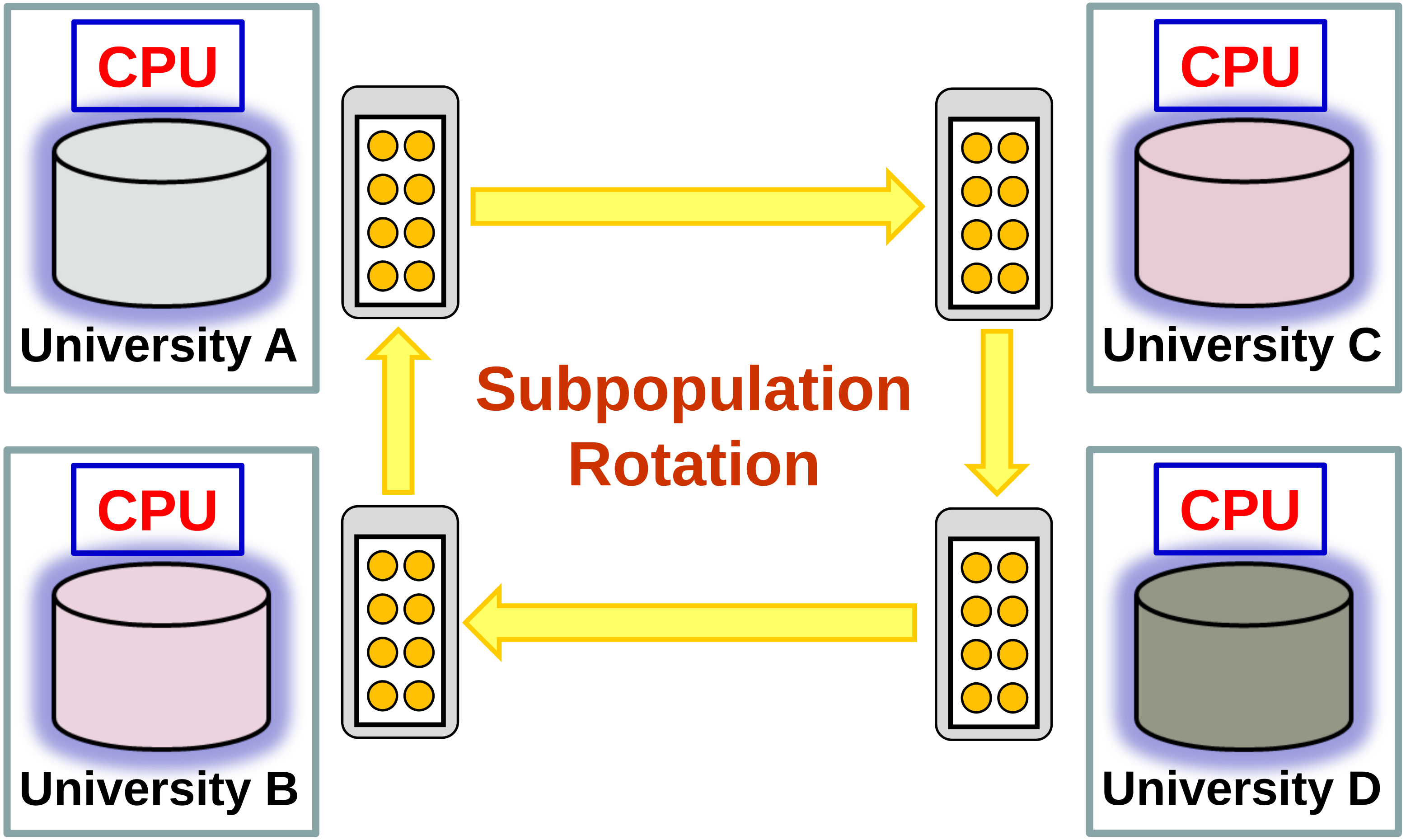
Conclusion

6. This model can be also used for learning from changing data bases.



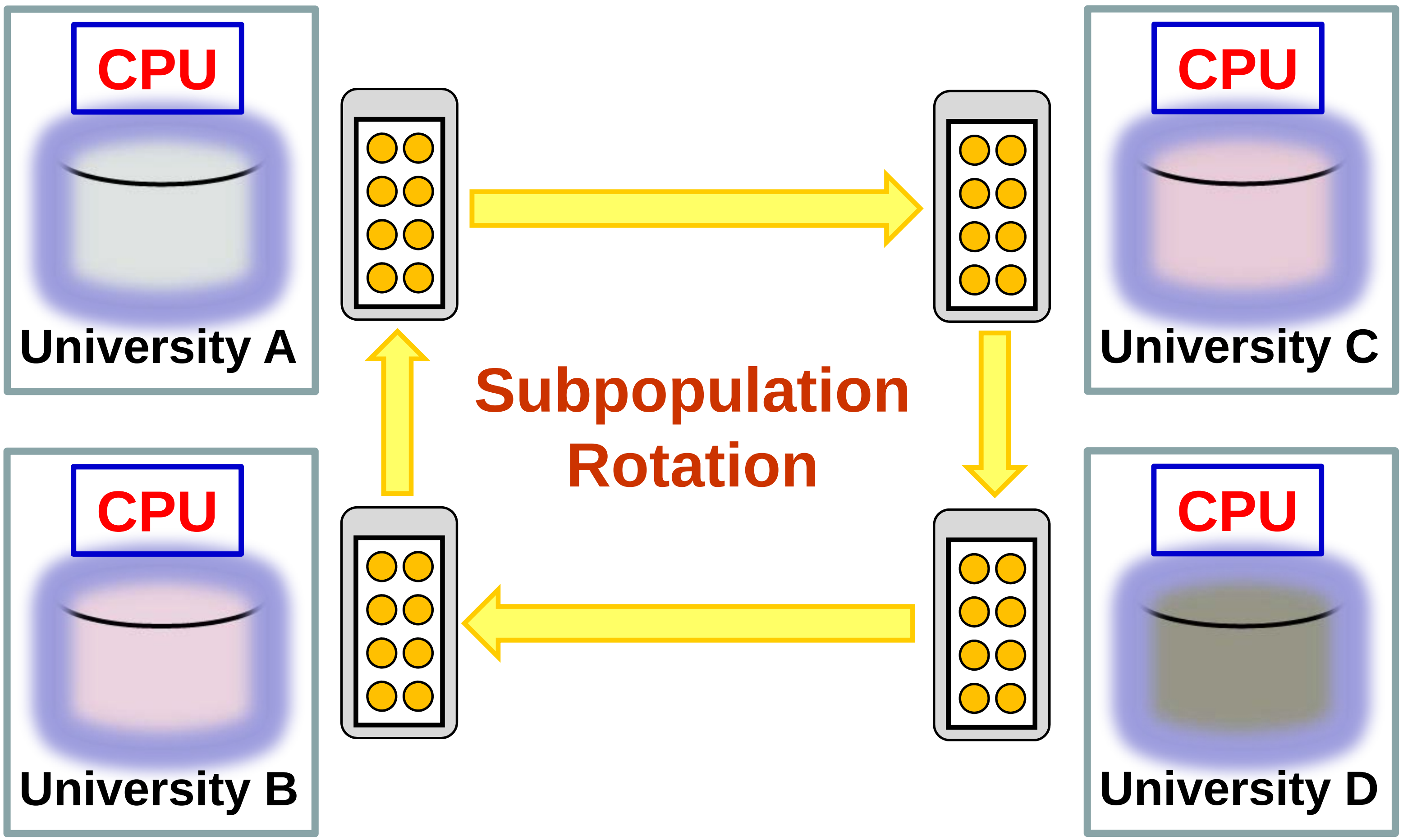
Conclusion

6. This model can be also used for learning from changing data bases.



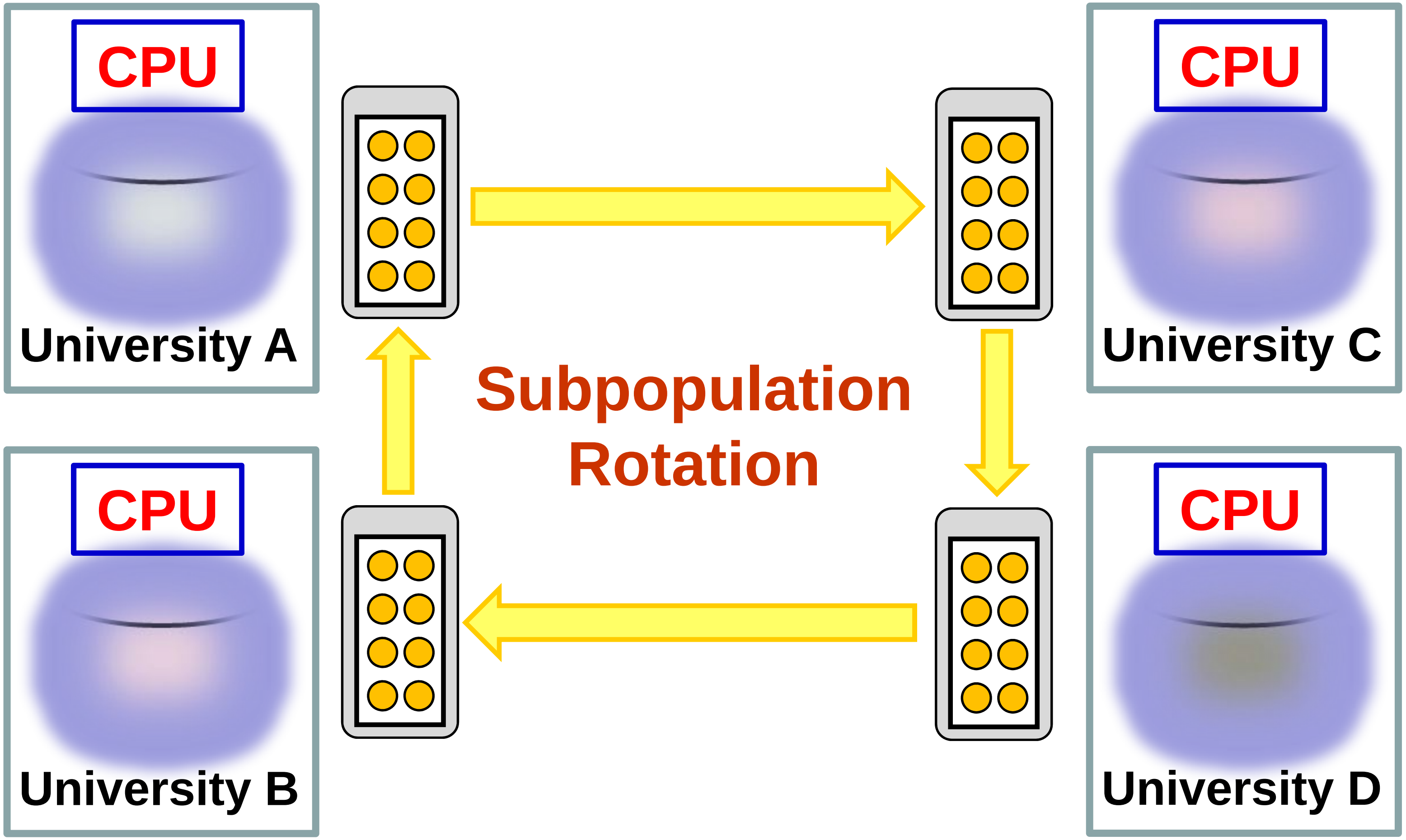
Conclusion

6. This model can be also used for learning from changing data bases.



Conclusion

6. This model can be also used for learning from changing data bases.



Conclusion

Thank you very much!