Speeding up of the Nelder-Mead Method by Data-driven Speculative Execution

Shuhei Watanabe Yoshihiko Ozaki Yoshiaki Bando Masaki Onishi

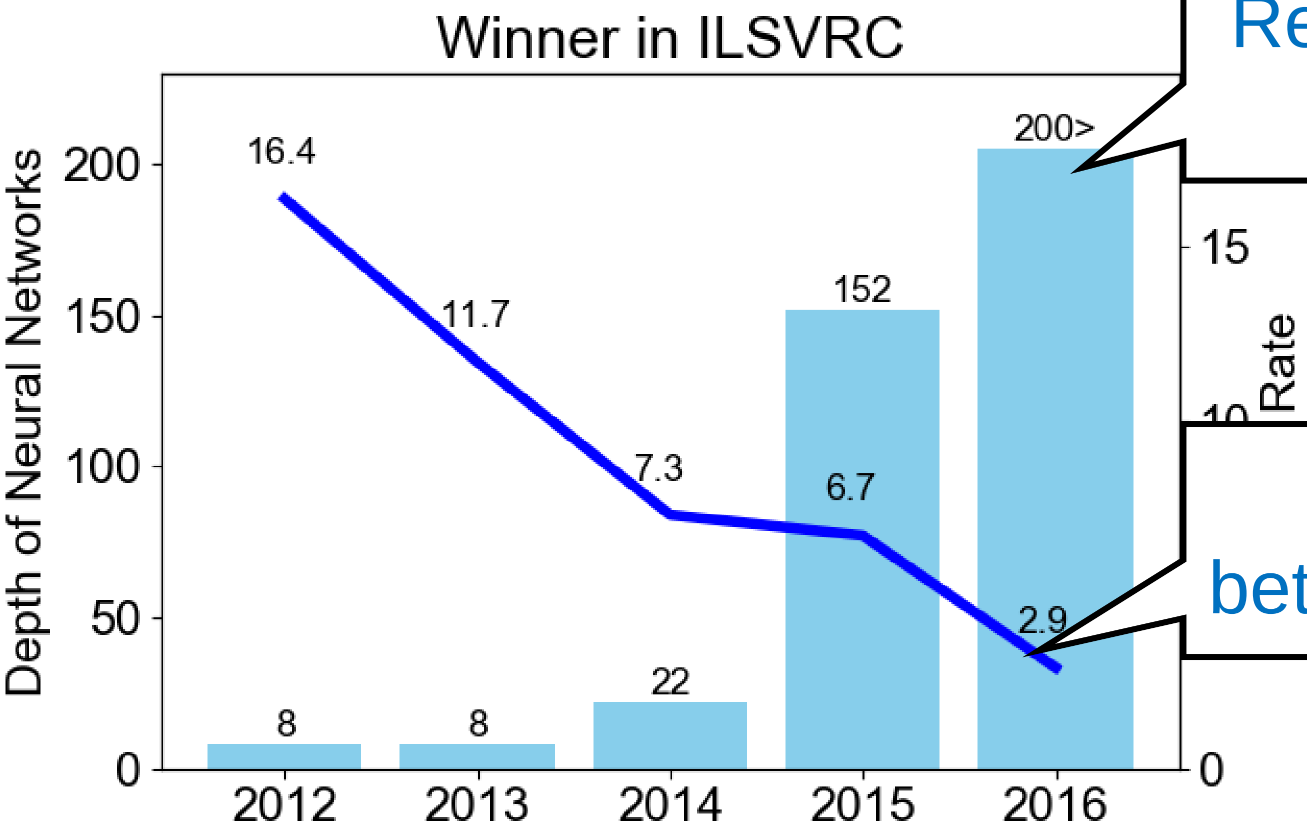
National Institute of Advanced Industrial Science and Technology

NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)

Background | Hyperparameter Optimization (HPO)

Fast **HPO** is important to use complex algorithms properly

- The hyperparameter (**HP**) space becomes bigger exponentially
- The performance significantly depends on HP settings



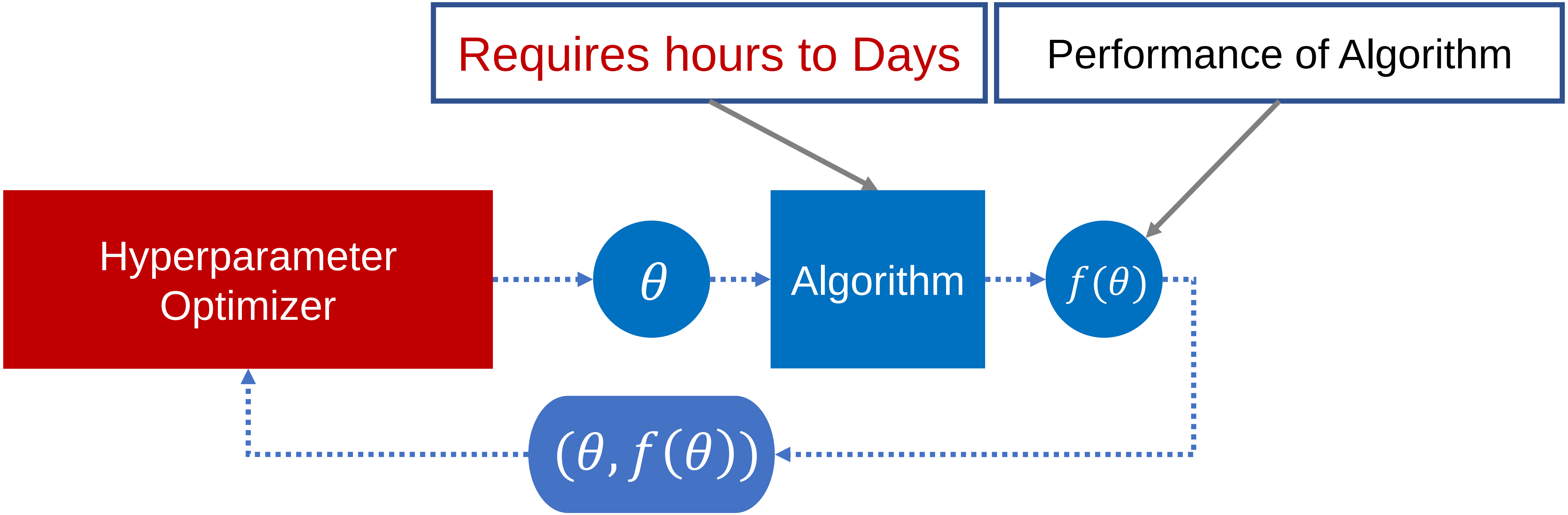
Requires Efficient HPO

HPO helps better performance

Background | Problem Setting of HPO

The goal of HPO is to find HP $\boldsymbol{\theta}$ maximizing the $\boldsymbol{f(\theta)}$

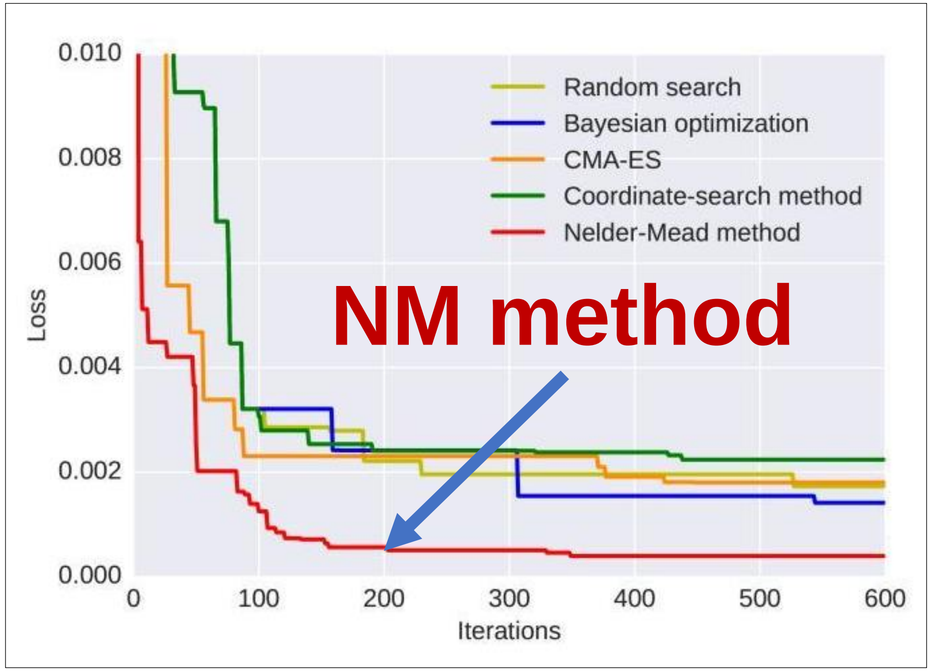
- θ is an HP setting of an algorithm
- The performance of the algorithm $f(\theta)$ is **black-box** and **expensive** to evaluate

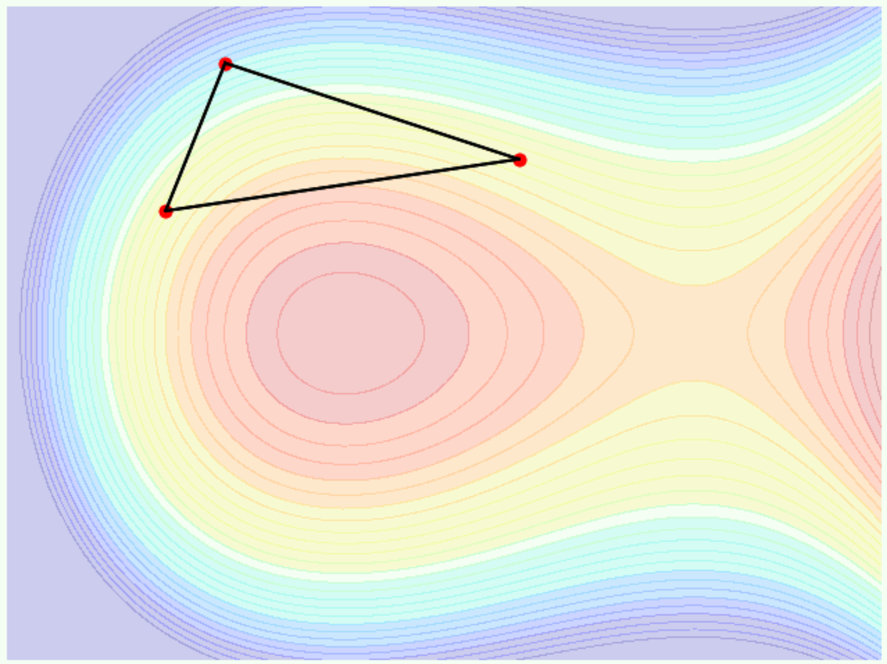


Related Work | HPO Methods

Nelder-Mead (NM) method converges faster [Ozaki+ 2017]

✓NM outperforms Bayesian optimization and CMA-ES on HPO of CNN





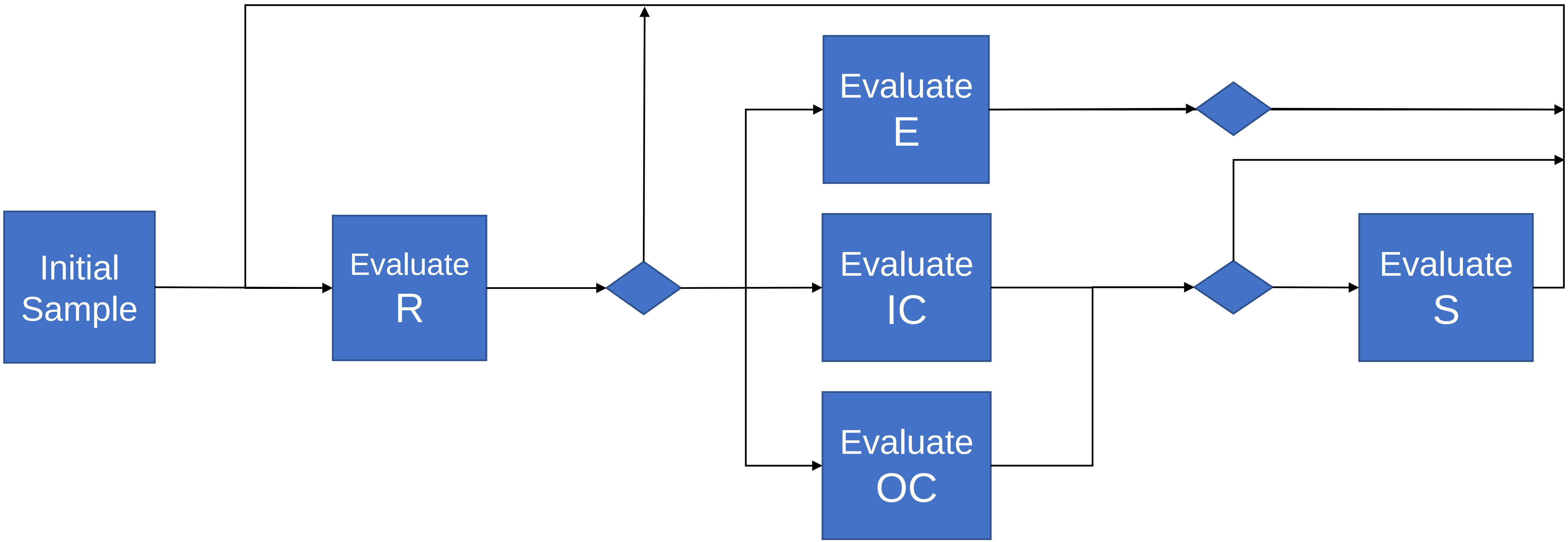
✗ NM is **sequential** and HPO of CNN requires several months

Therefore, we propose **parallel method** for the NM method

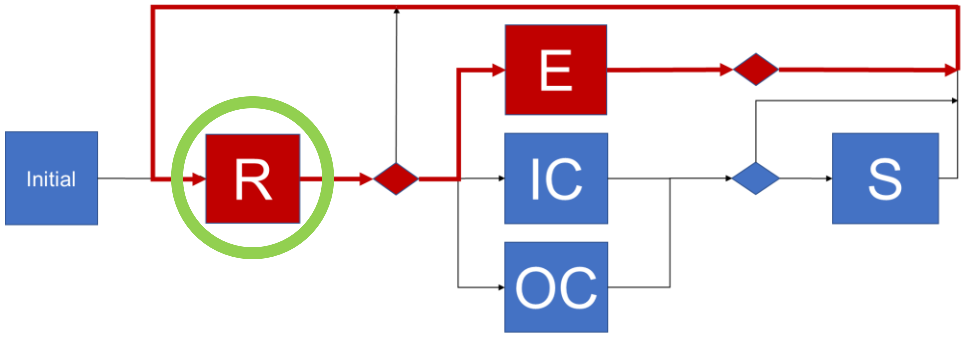
NM Method | Flowchart

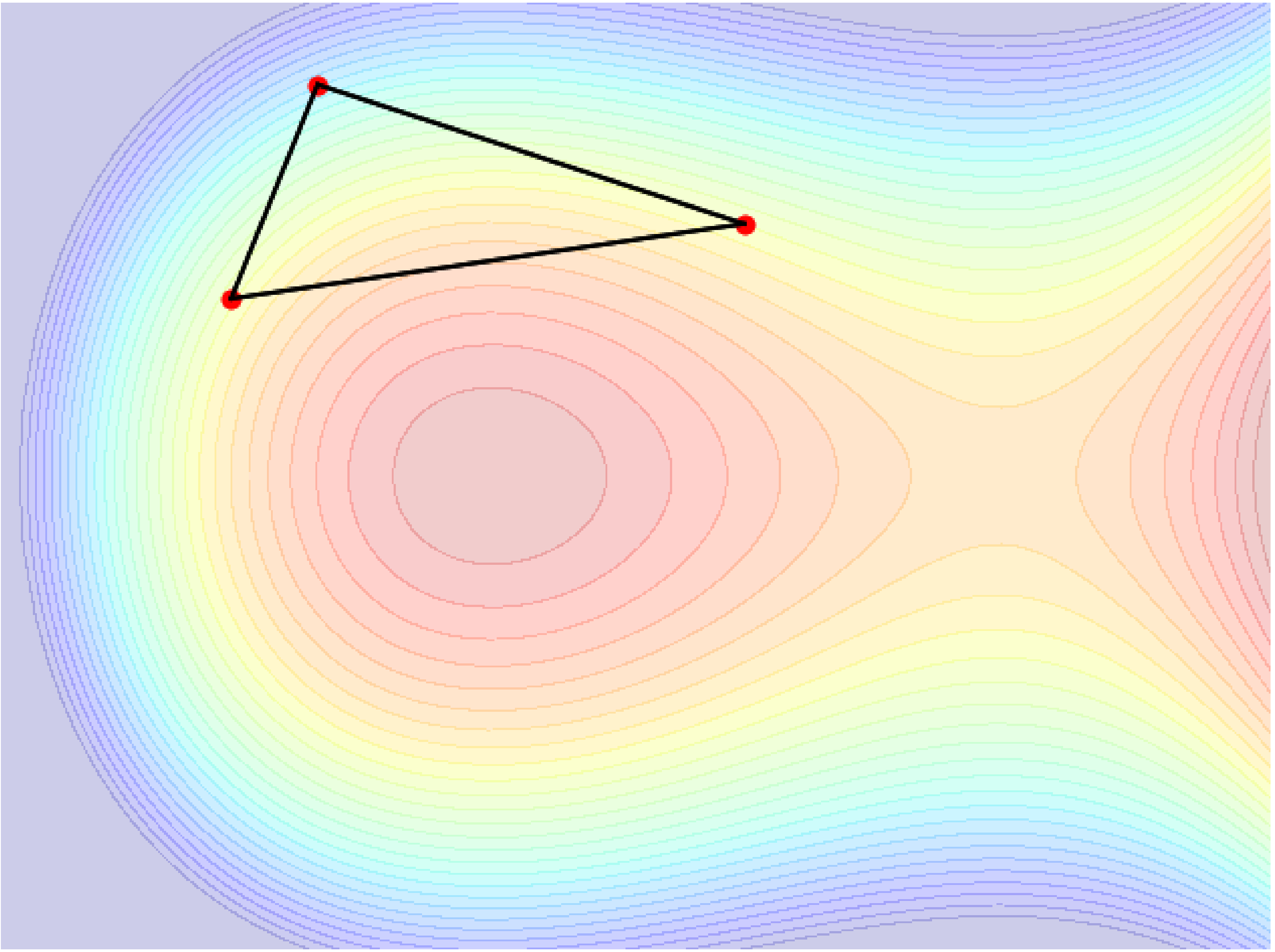
Iterates the operations below until the termination

- There are **5 operations** and **6 possible transition** in an iteration



NM Method | Possible Transitions 1 ~part 1~

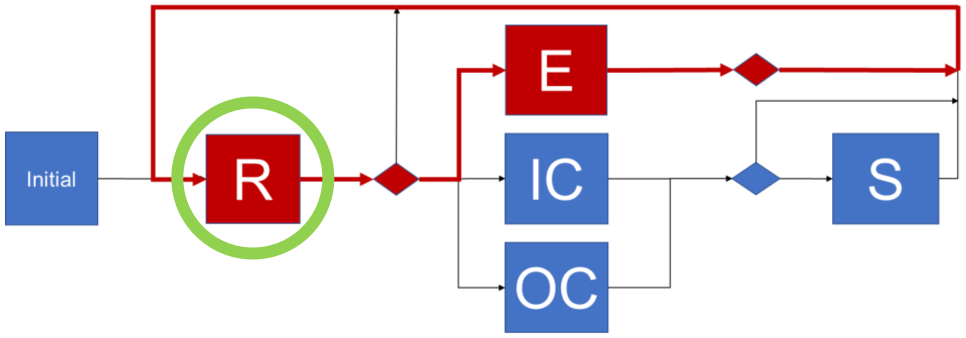




Total: 2 Evaluation Time

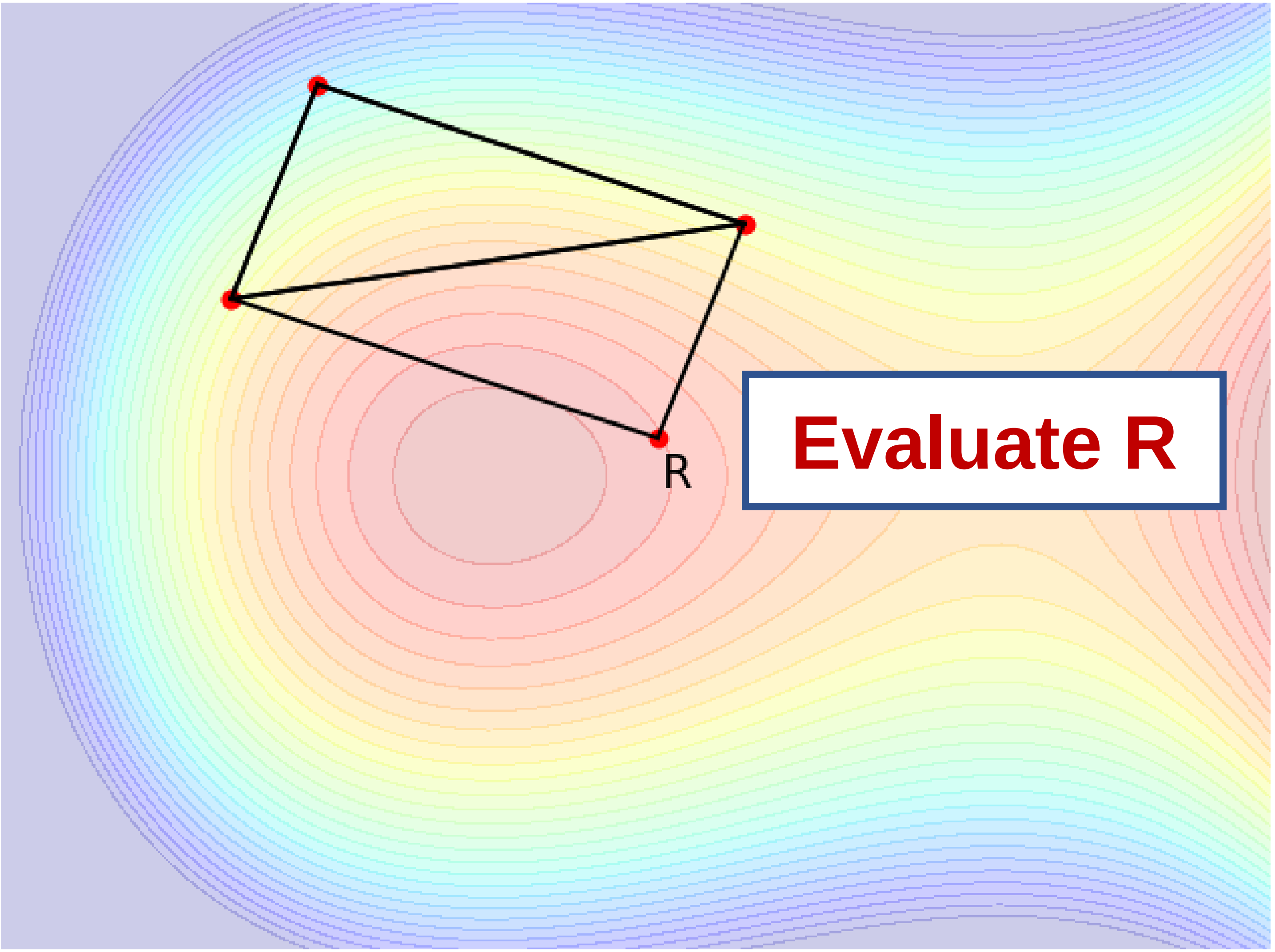
1. **Evaluate R**
2. If R is the best
3. Evaluate E
4. Take the better, R or E

NM Method | Possible Transitions 1 ~part 2~

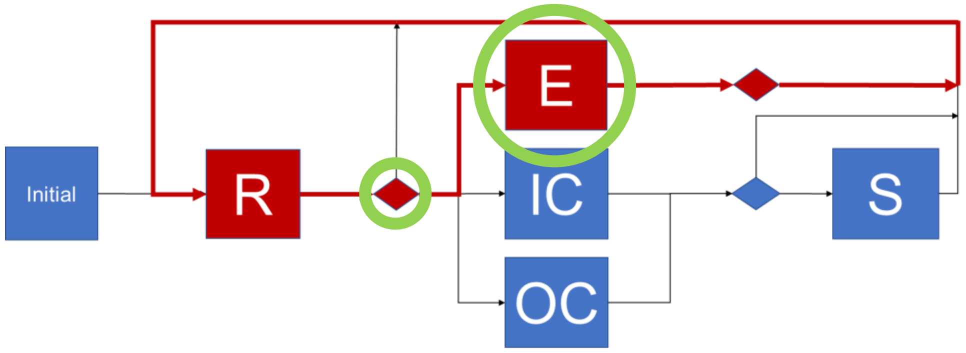


Total: 2 Evaluation Time

1. **Evaluate R**
2. If R is the best
3. Evaluate E
4. Take the better, R or E

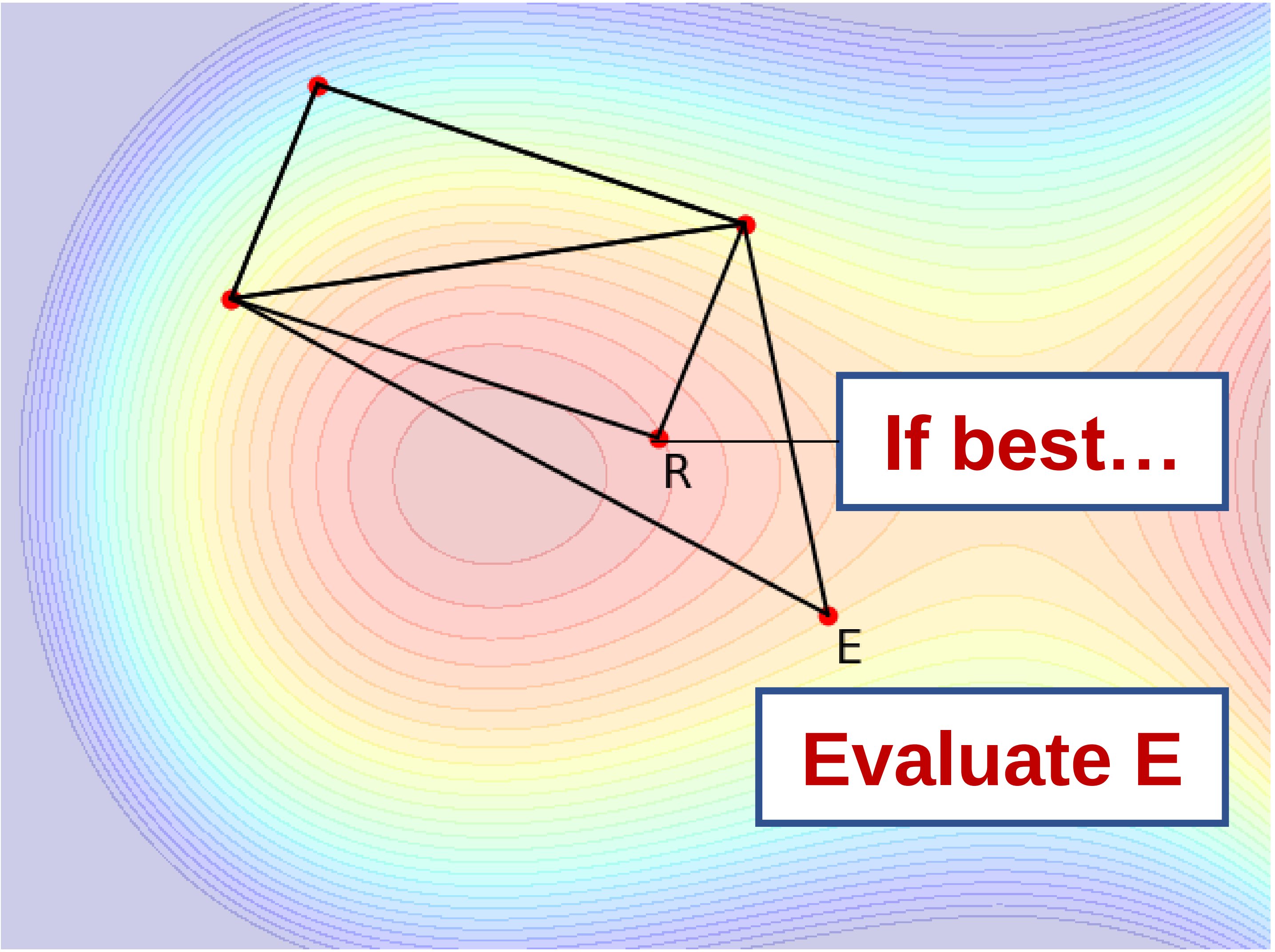


NM Method | Possible Transitions 1 ~part 3~

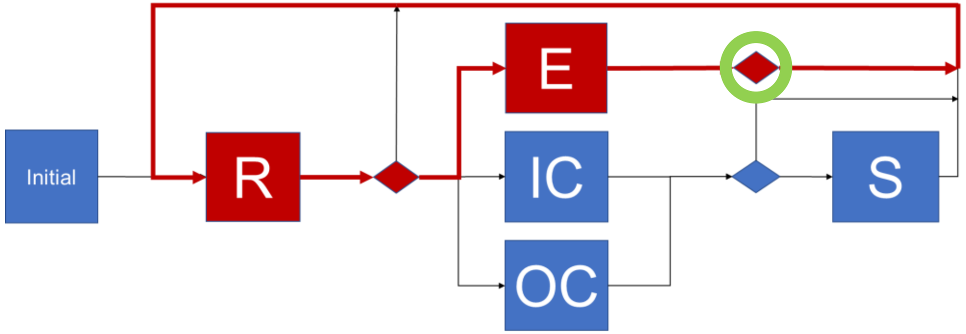


Total: 2 Evaluation Time

1. Evaluate R
2. **If R is the best**
3. **Evaluate E**
4. Take the better, R or E

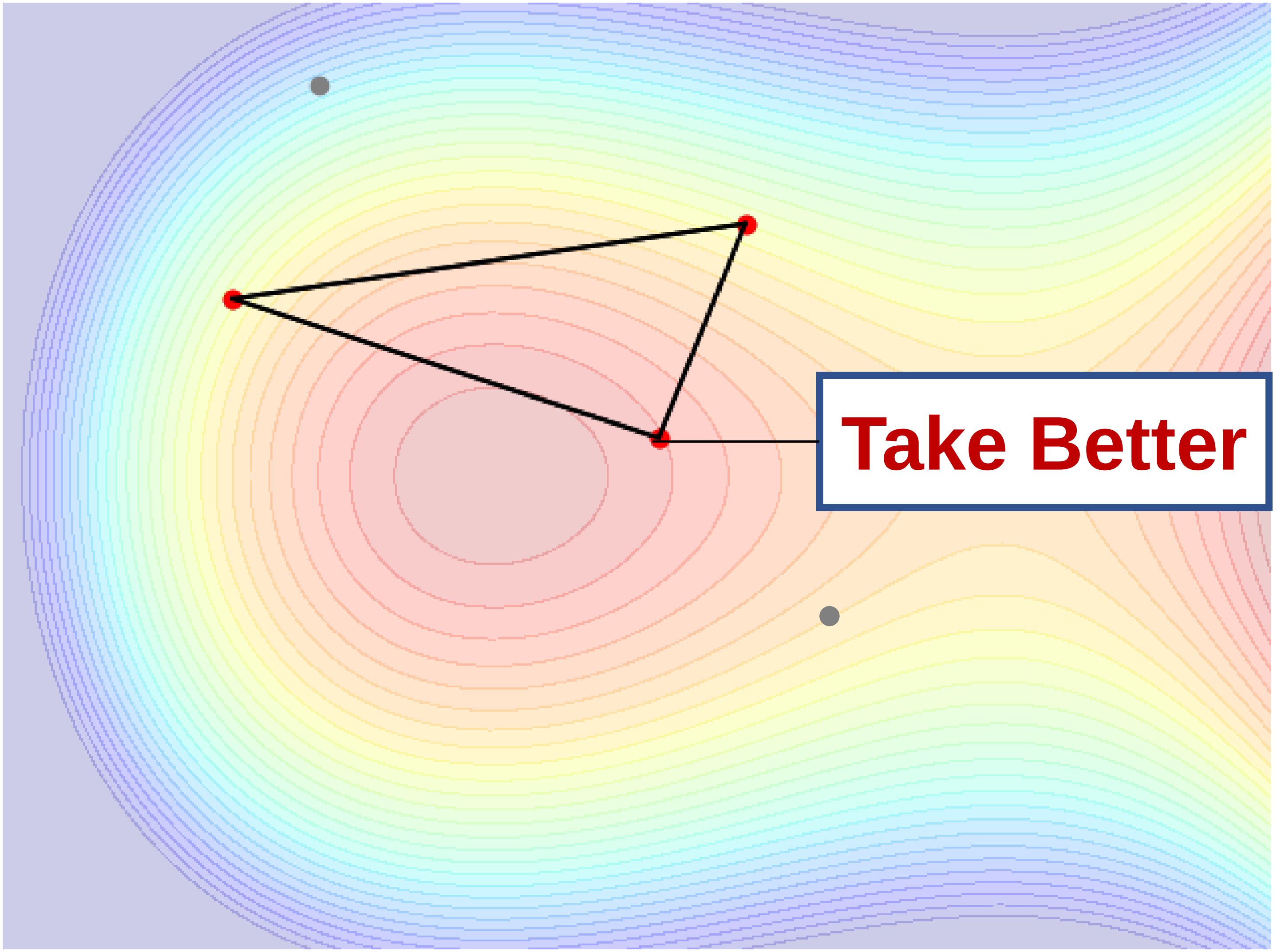


NM Method | Possible Transitions 1 ~part 4~

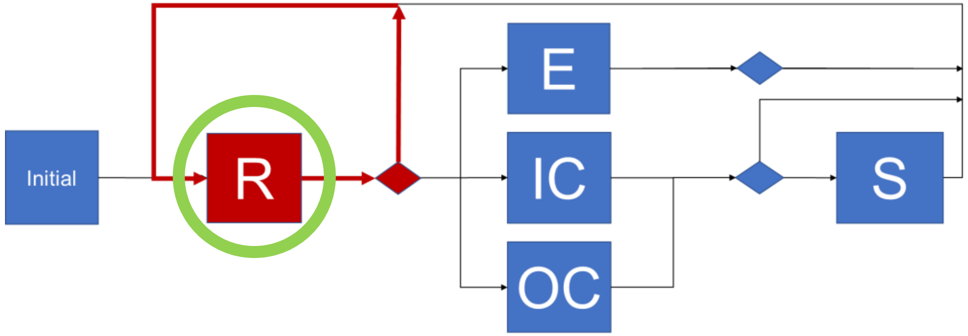


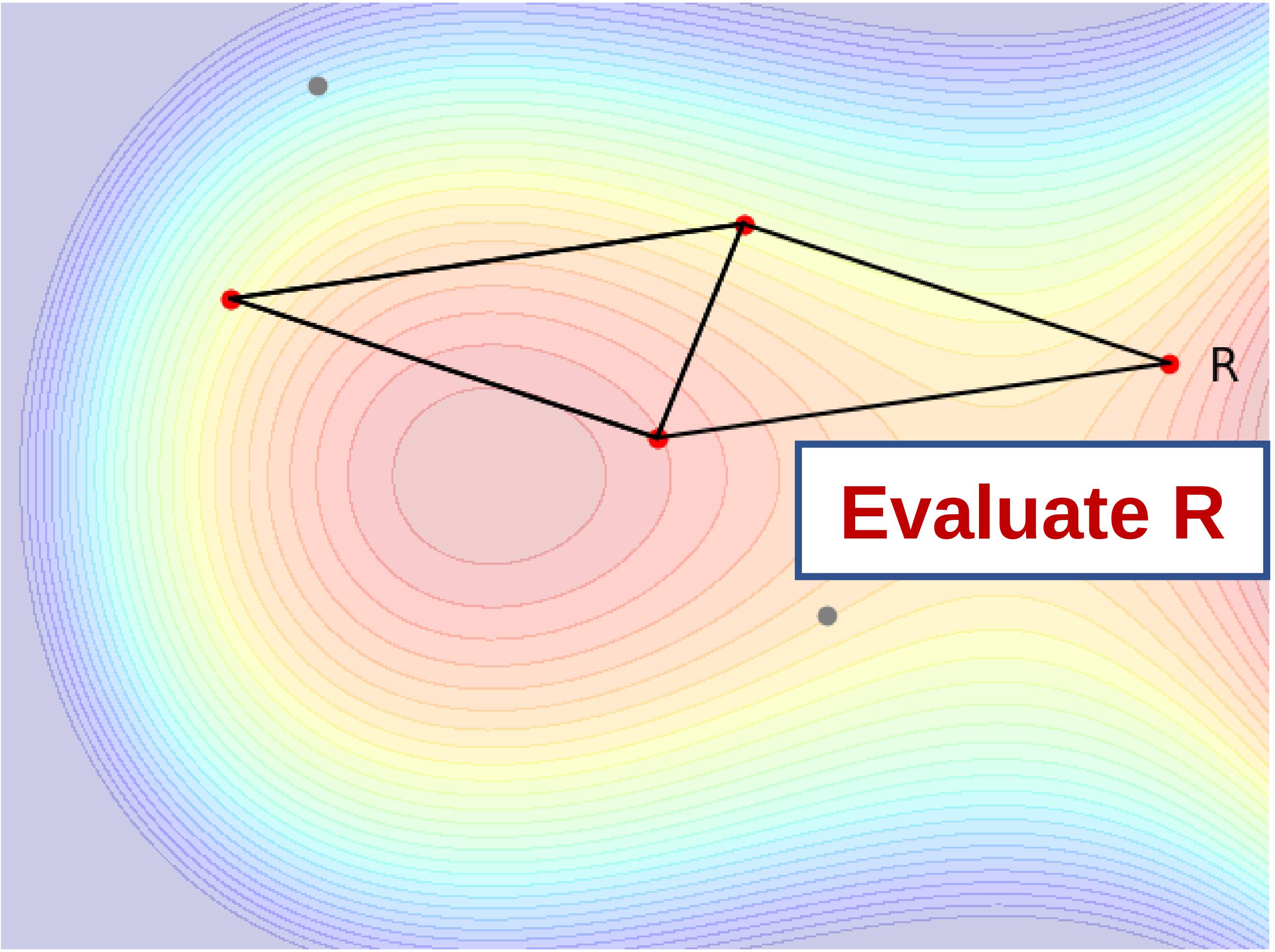
Total: 2 Evaluation Time

1. Evaluate R
2. If R is the best
3. Evaluate E
4. **Take the better, R or E**



NM Method | Possible Transitions 2 ~part 1~

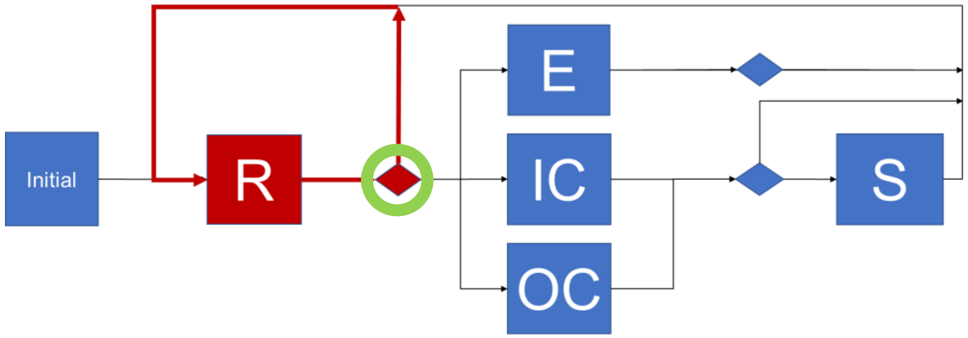




Total: 1 Evaluation Time

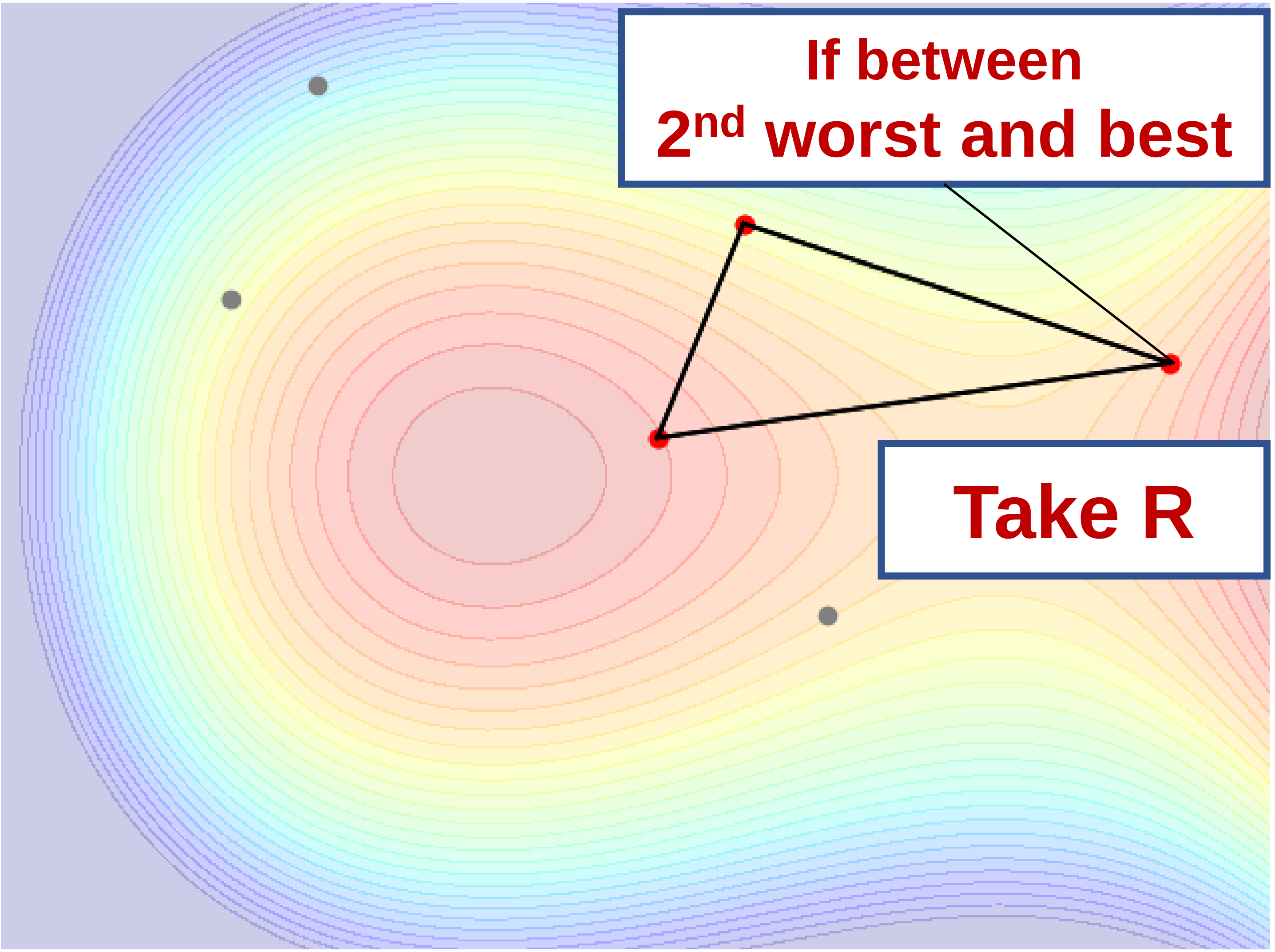
1. **Evaluate R**
2. If R is between best and 2nd worst
3. Accepting R

NM Method | Possible Transitions 2 ~part 2~

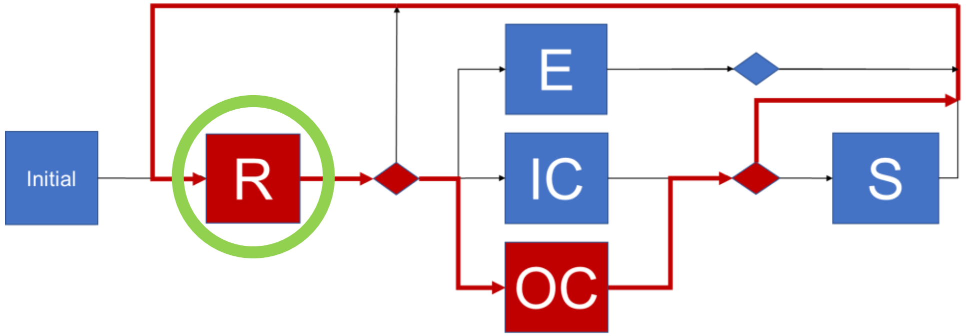


Total: 1 Evaluation Time

1. Evaluate R
2. **If R is between best and 2nd worst**
3. **Accepting R**

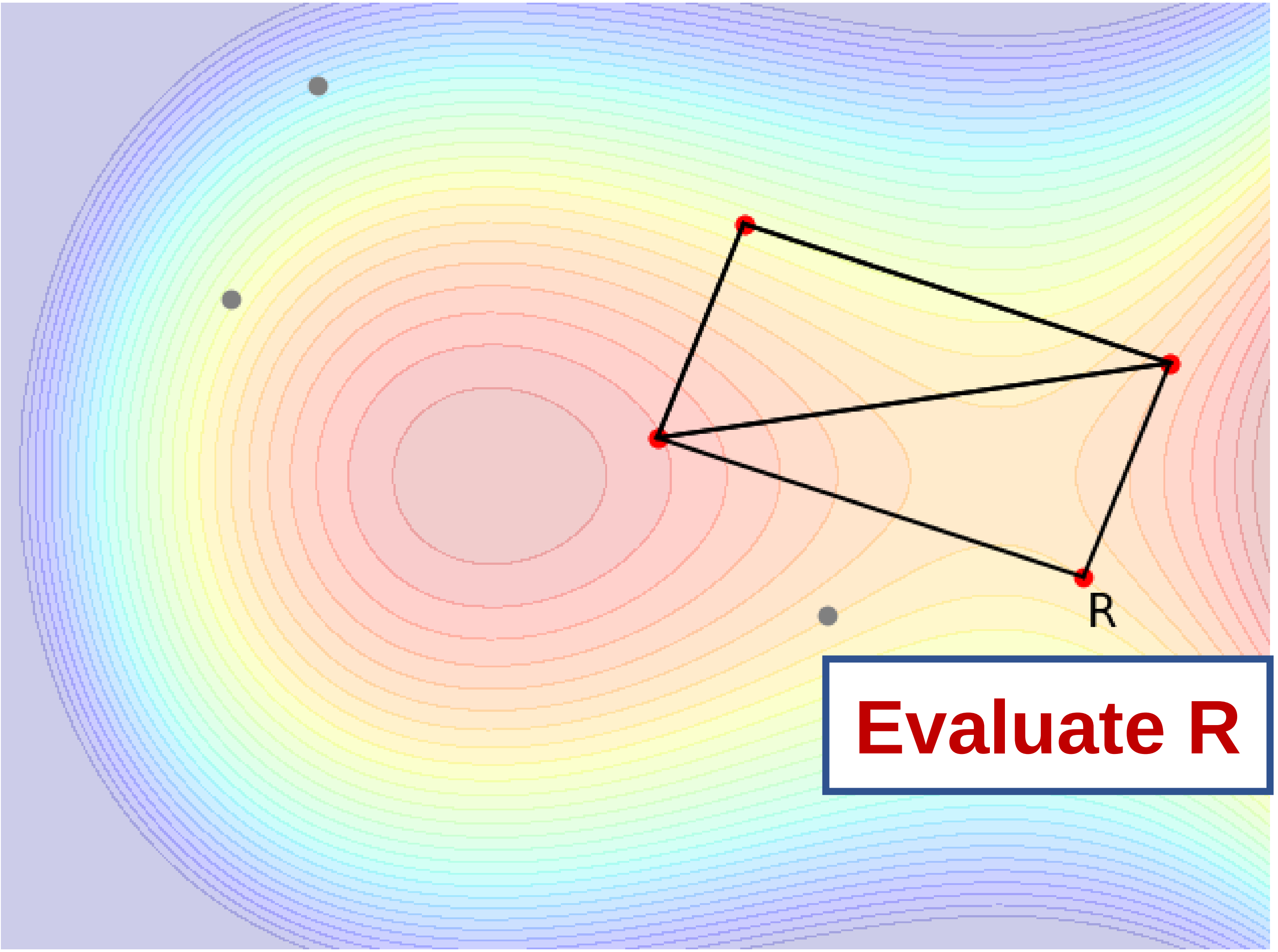


NM Method | Possible Transitions 3 ~part 1~

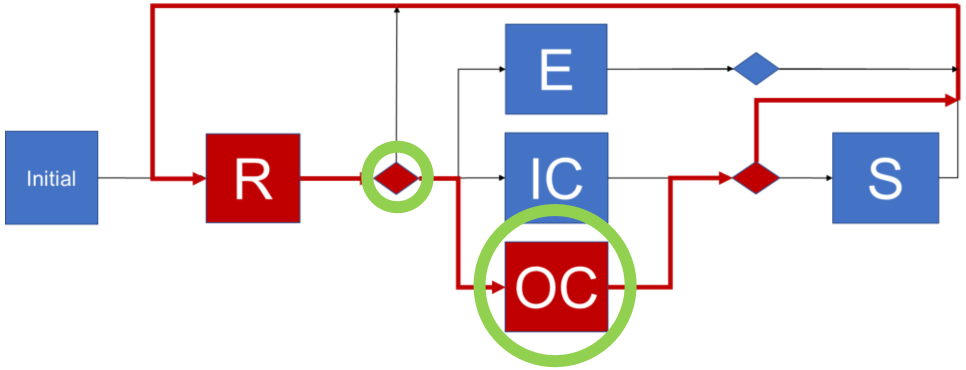


Total: 2 Evaluation Time

1. **Evaluate R**
2. If R is the 2nd worst
3. Evaluate OC
4. If OC is better than R take OC

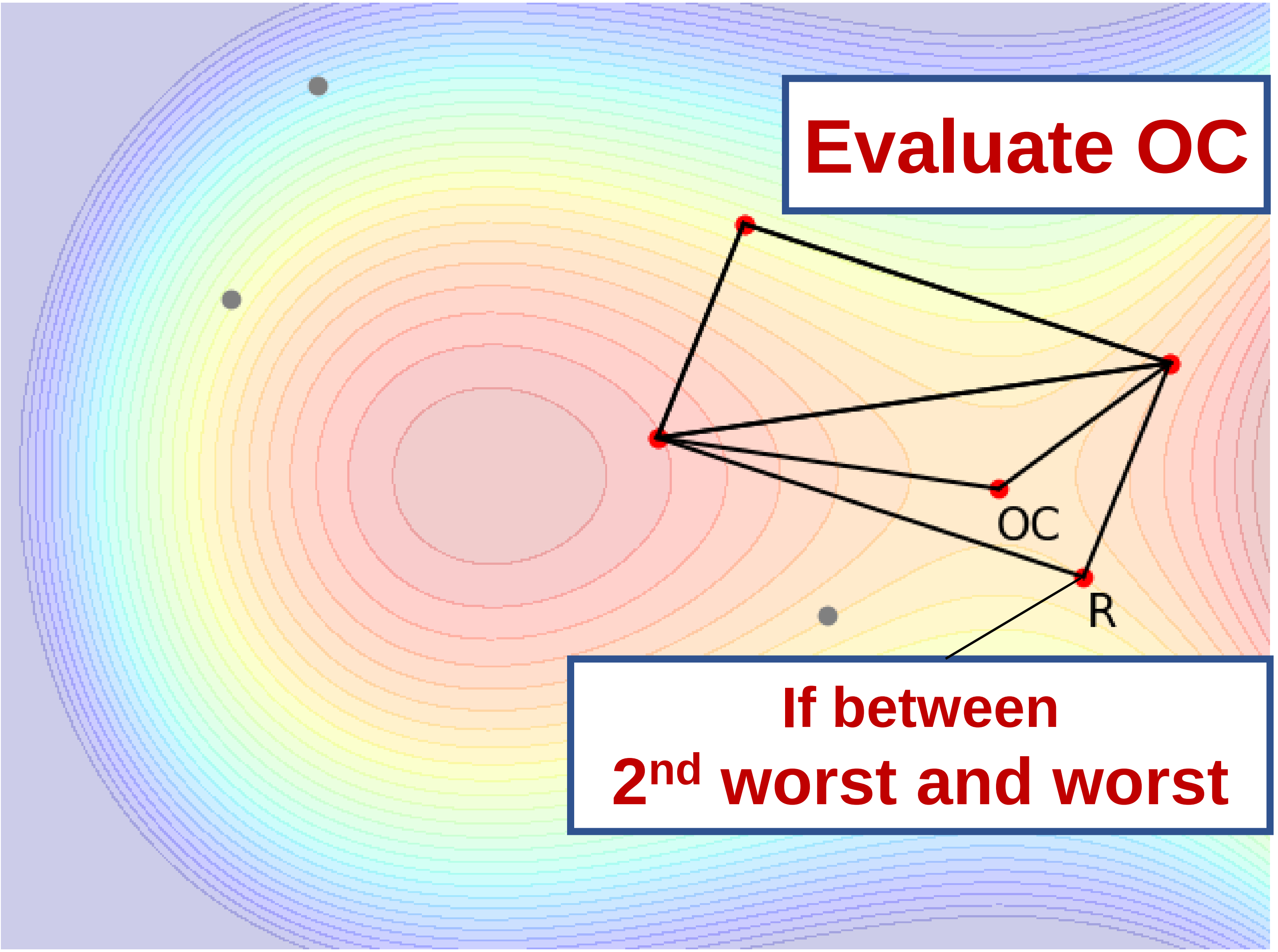


NM Method | Possible Transitions 3 ~part 2~

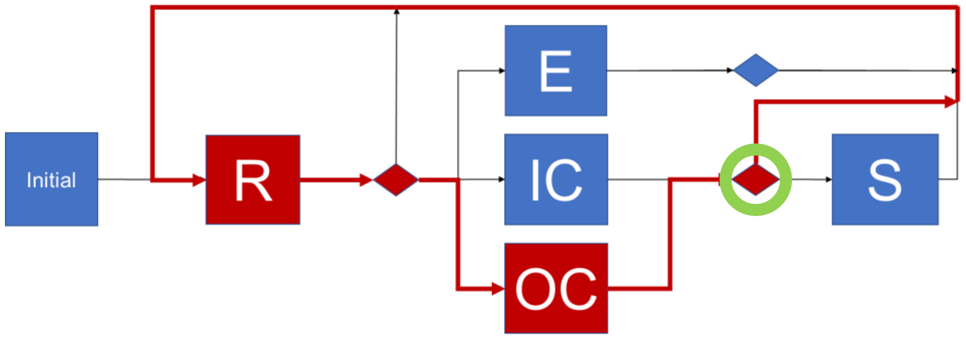


Total: 2 Evaluation Time

1. Evaluate R
2. **If R is the 2nd worst**
3. **Evaluate OC**
4. If OC is better than R take OC

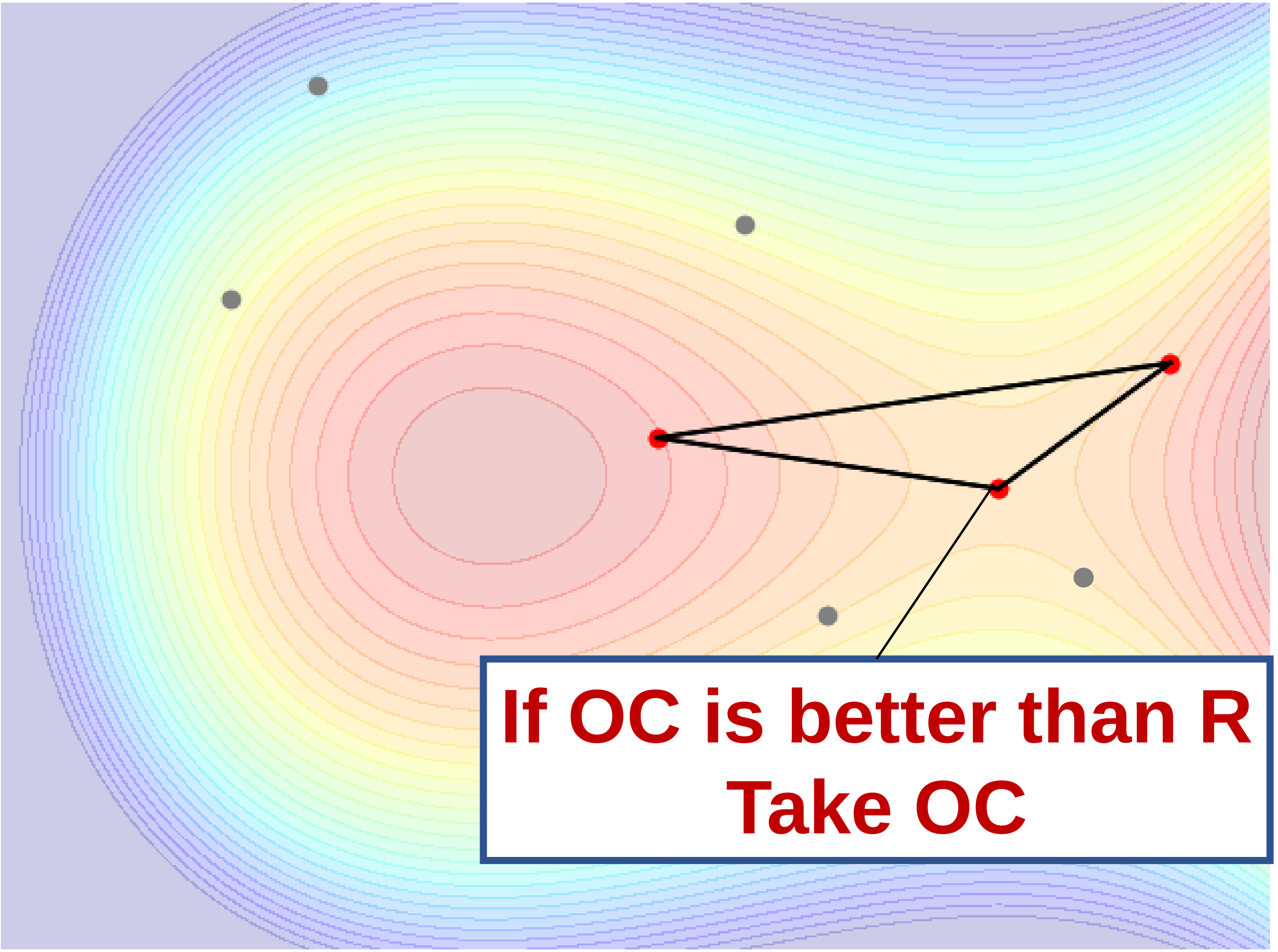


NM Method | Possible Transitions 3 ~part 3~

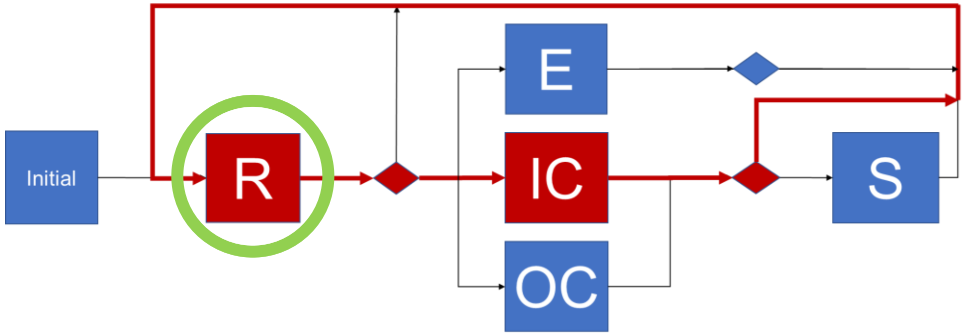


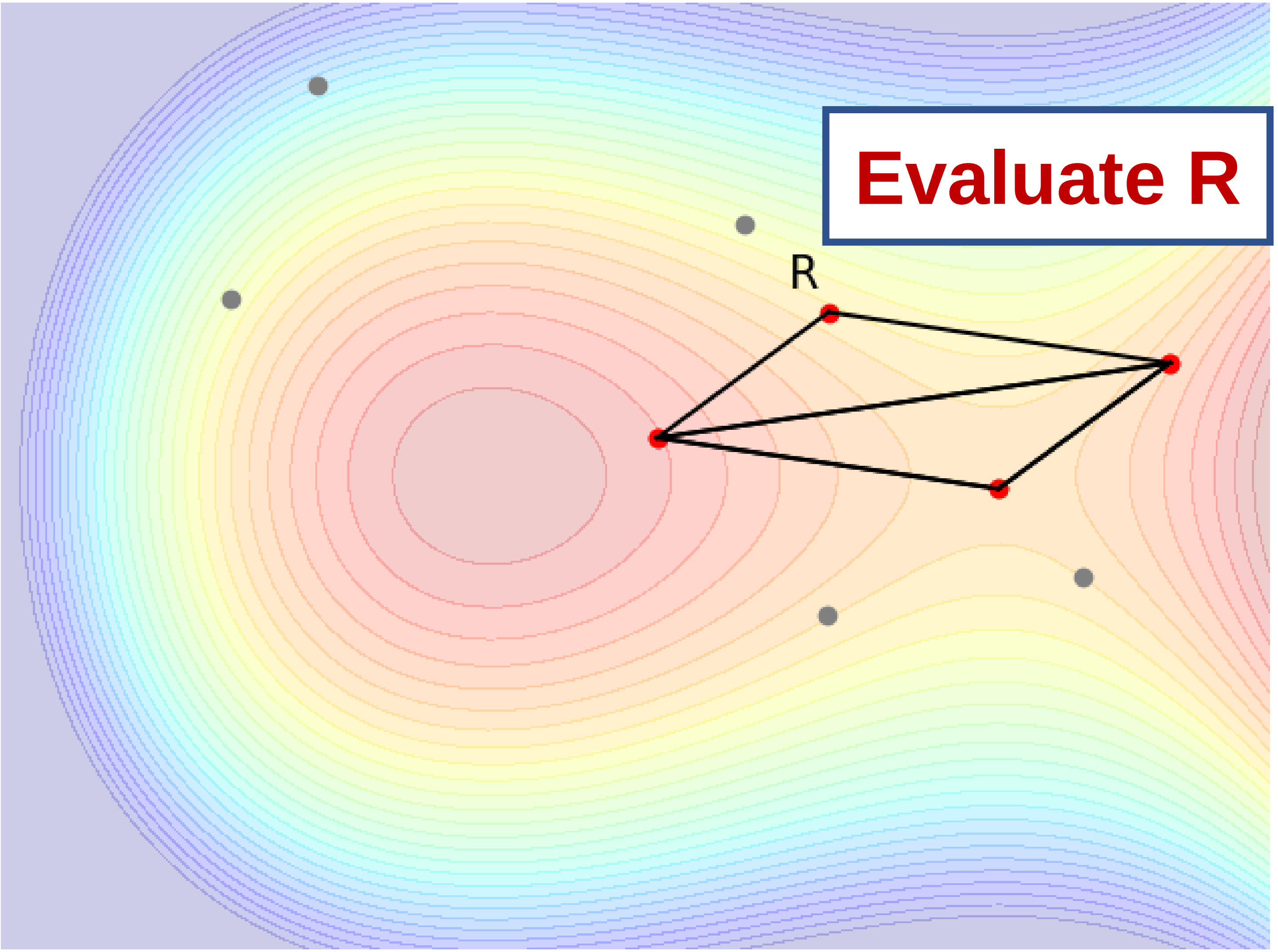
Total: 2 Evaluation Time

1. Evaluate R
2. If R is the 2nd worst
3. Evaluate OC
4. **If OC is better than R take OC**



NM Method | Possible Transitions 4 ~part 1~

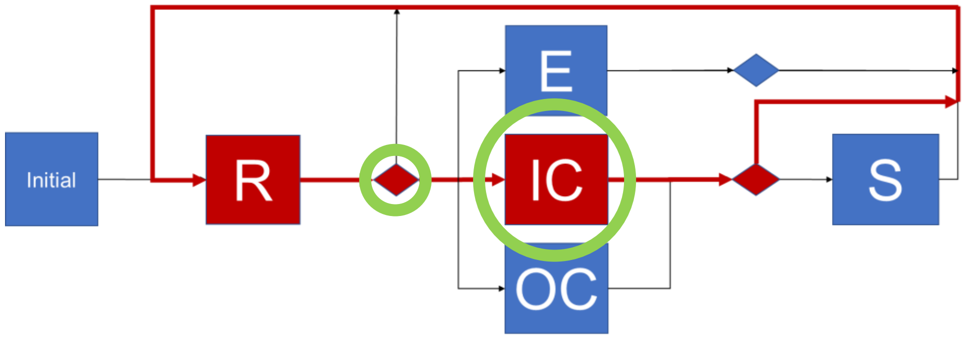




Total: 2 Evaluation Time

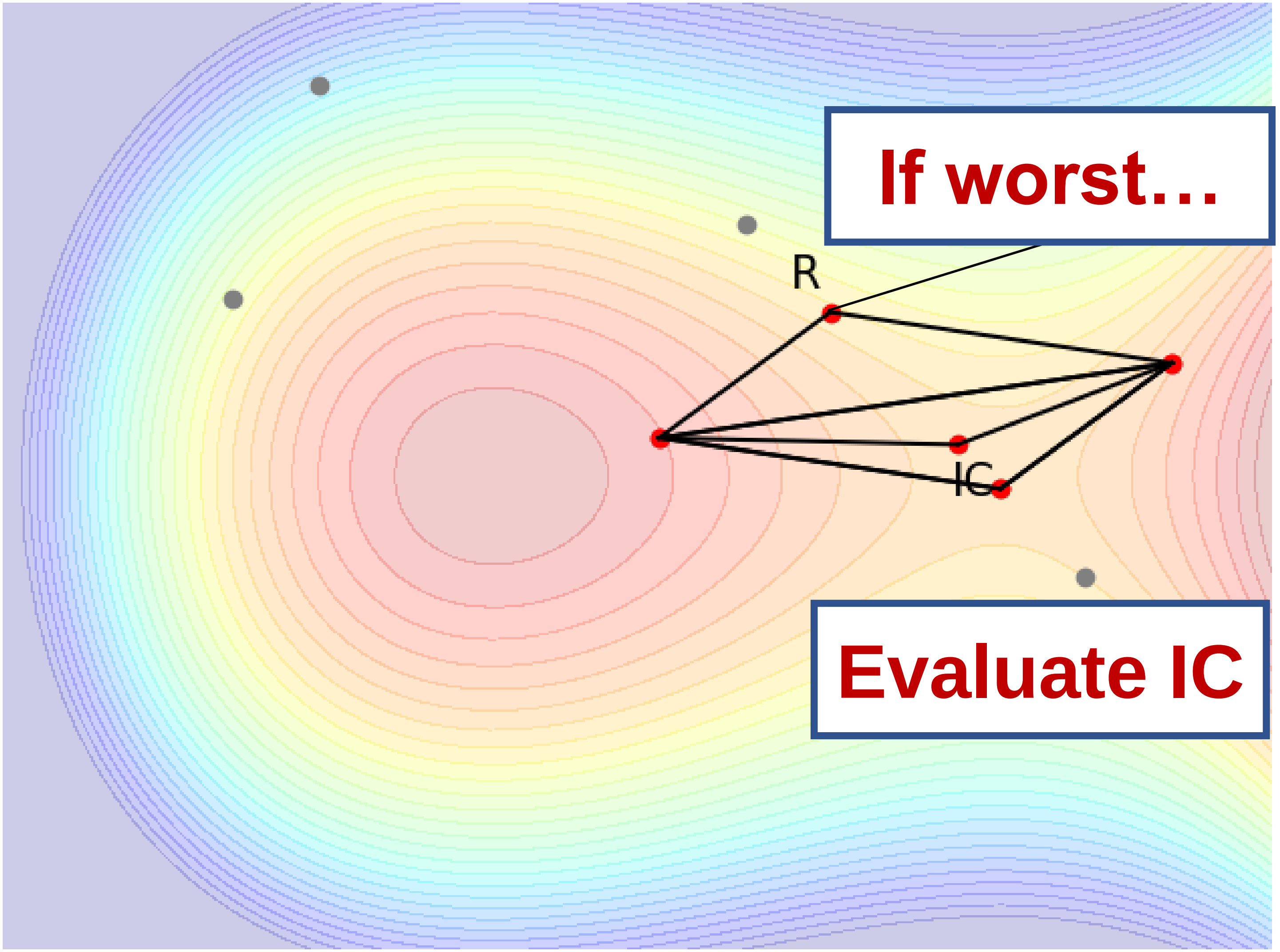
1. **Evaluate R**
2. If R is the worst
3. Evaluate IC
4. If IC is better than the worst take IC

NM Method | Possible Transitions 4 ~part 2~

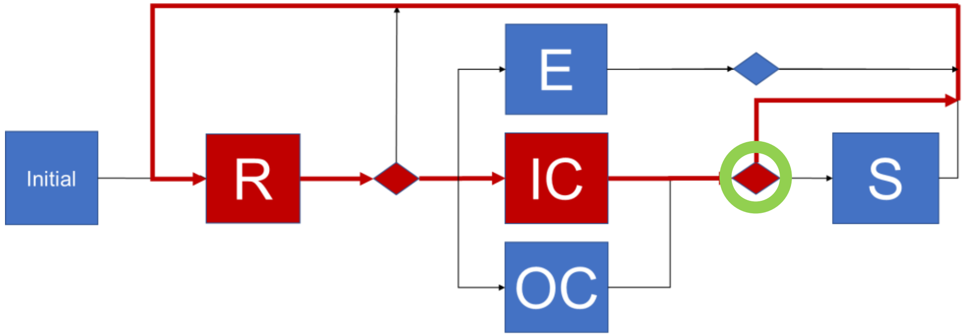


Total: 2 Evaluation Time

1. Evaluate R
2. **If R is the worst**
3. **Evaluate IC**
4. If IC is better than the worst take IC

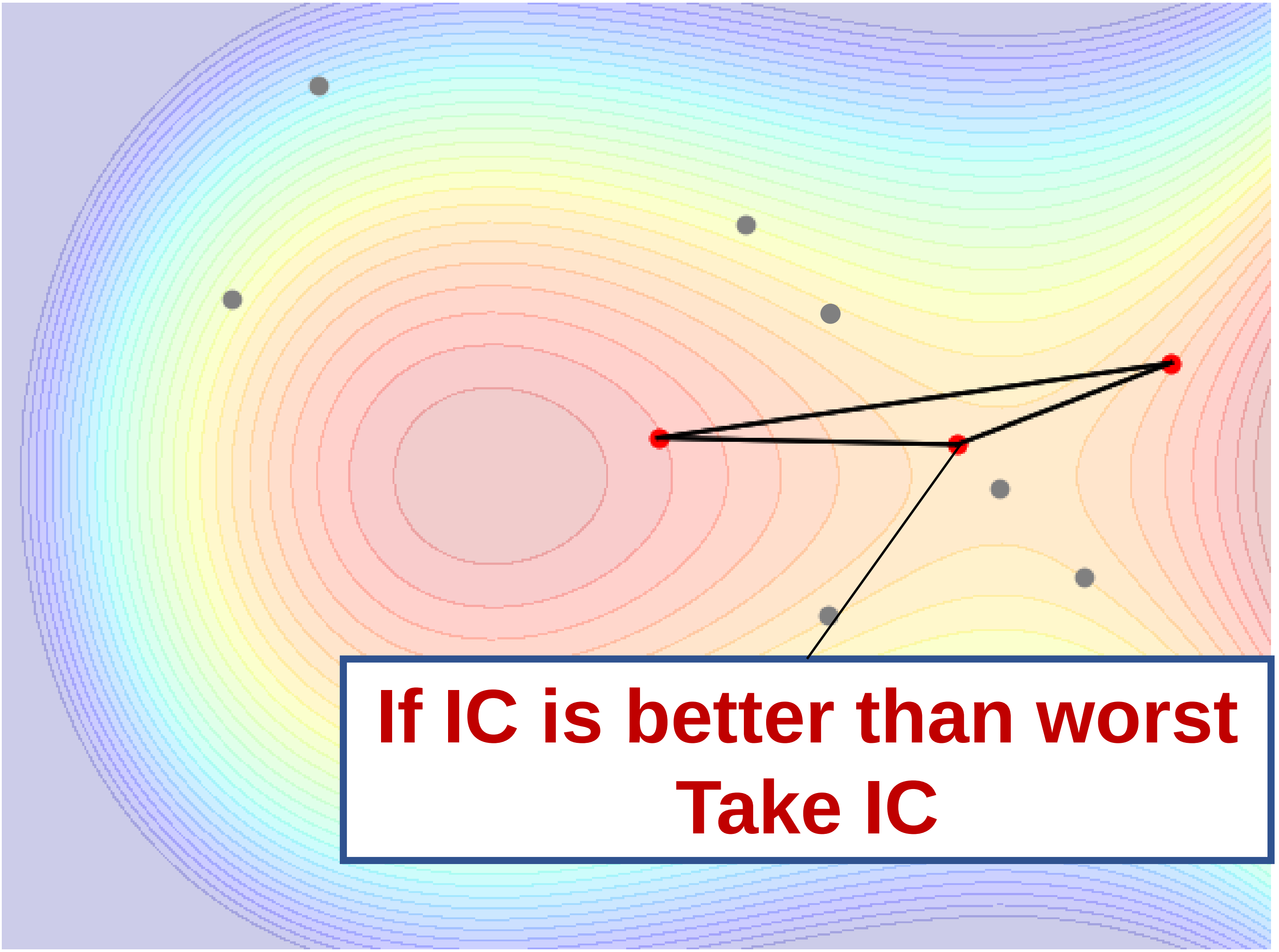


NM Method | Possible Transitions 4 ~part 3~

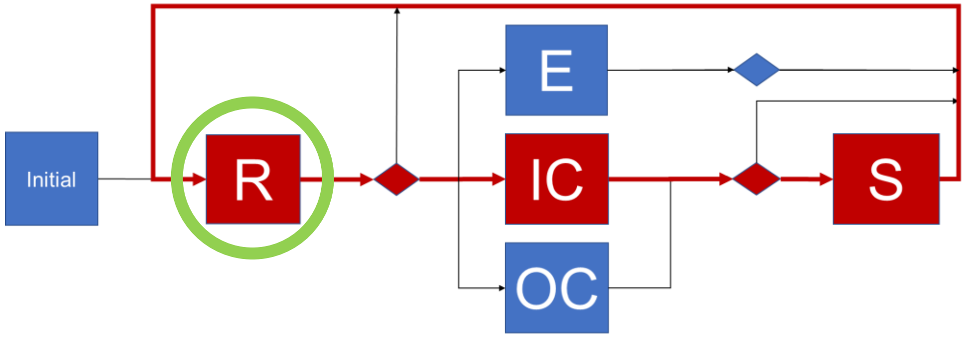


Total: 2 Evaluation Time

1. Evaluate R
2. If R is the worst
3. Evaluate IC
4. **If IC is better than the worst take IC**

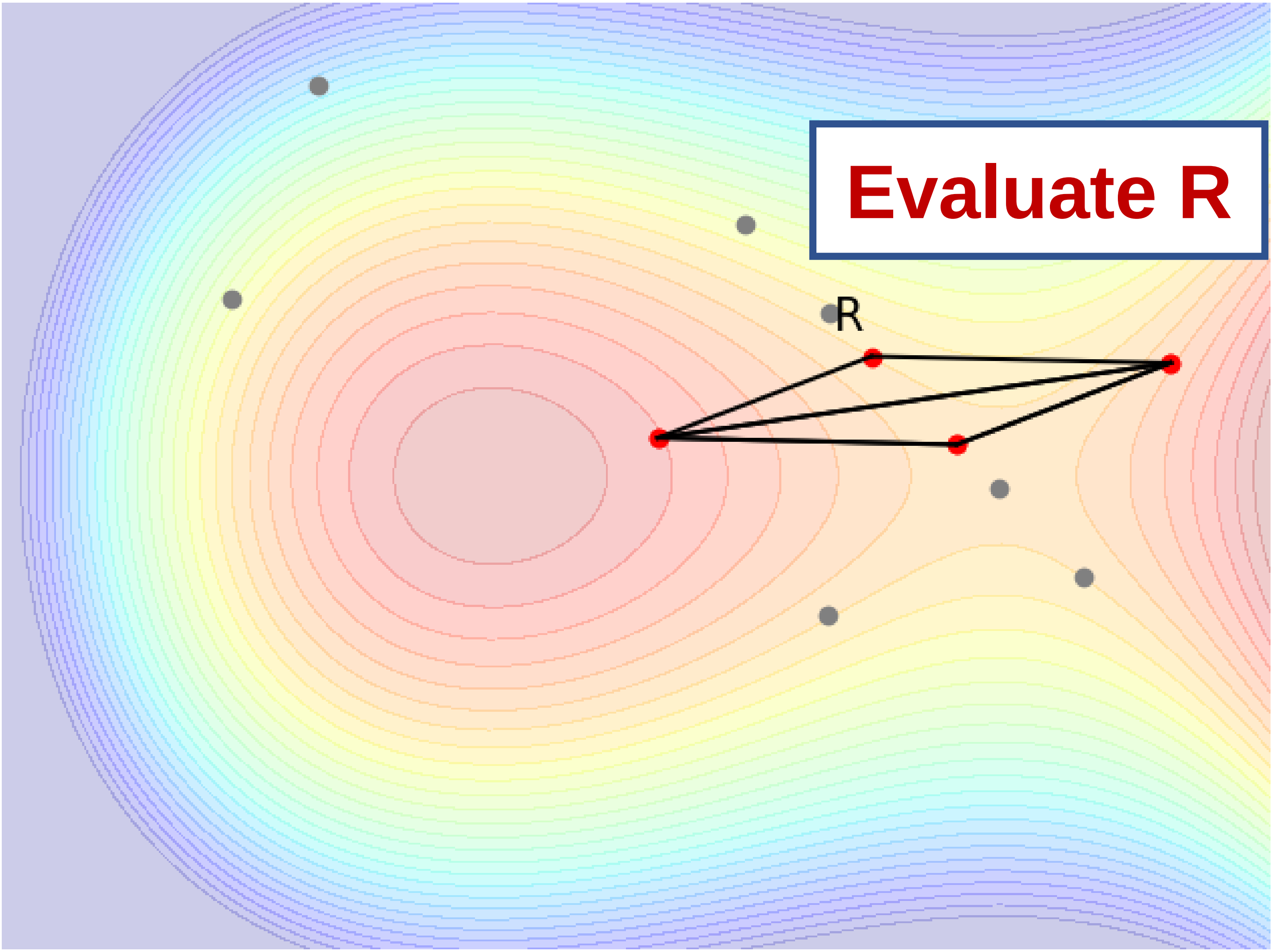


NM Method | Possible Transitions 5, 6 ~part 1~

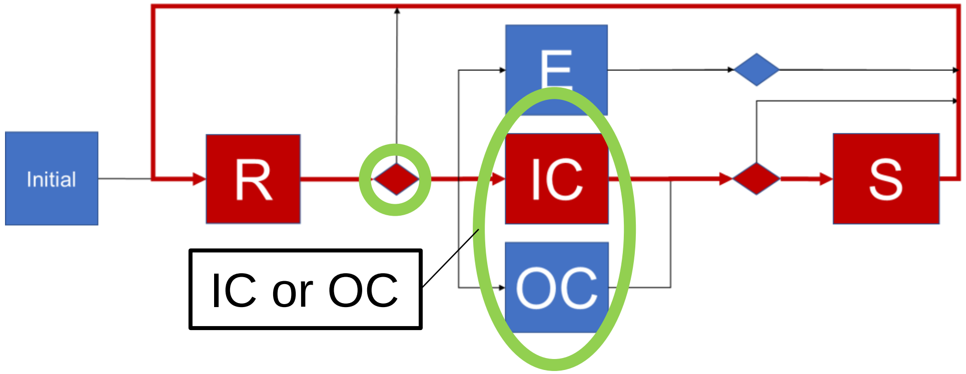


Total: 3 Evaluation Time

1. **Evaluate R**
2. If R is the worst or 2nd worst
3. Evaluate IC or OC
4. If IC or OC does not improve
5. Evaluate S

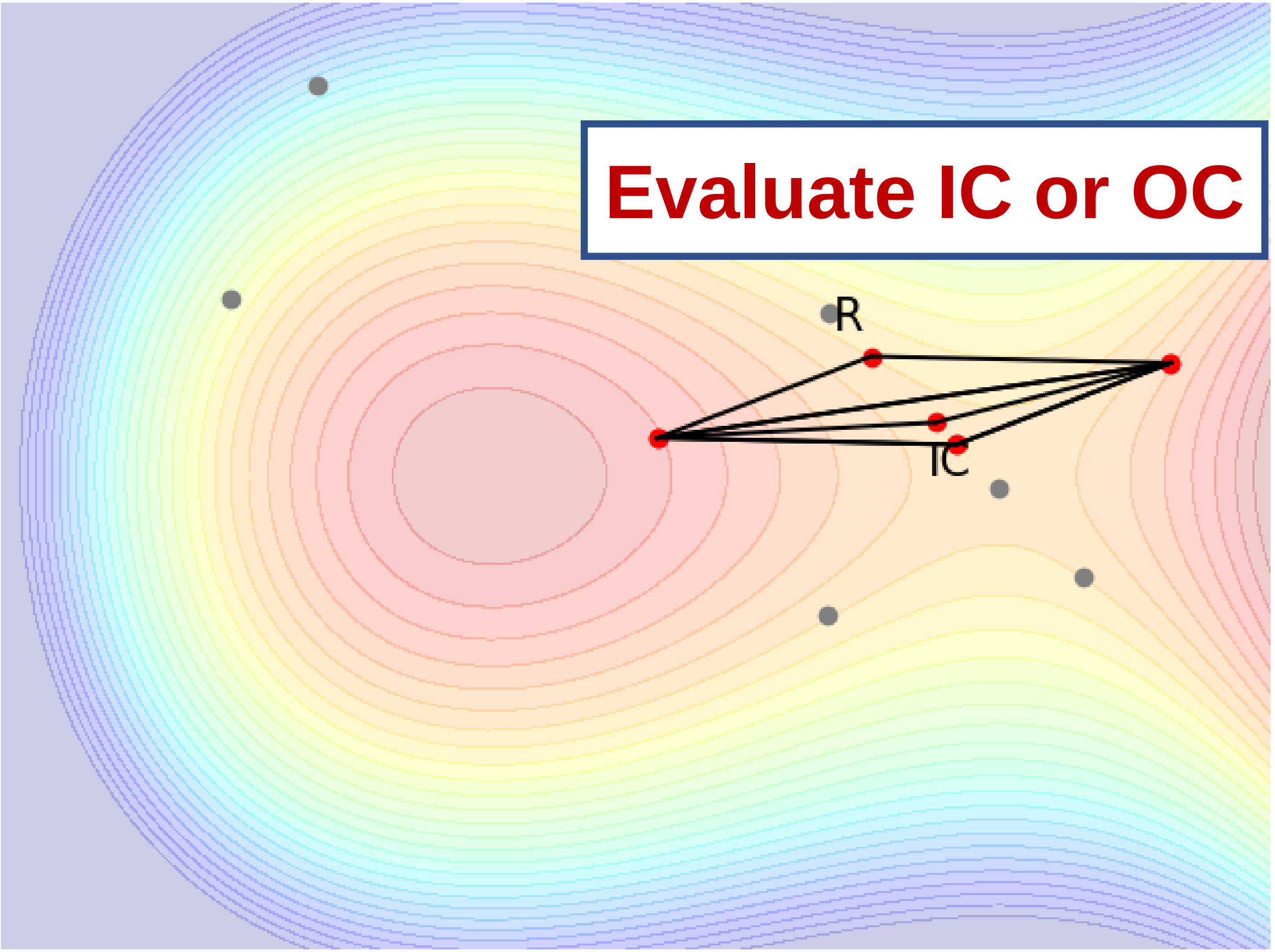


NM Method | Possible Transitions 5, 6 ~part 2~

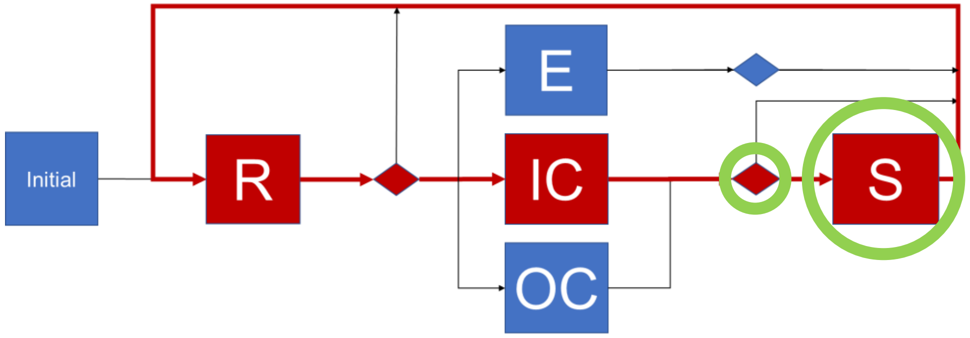


Total: 3 Evaluation Time

1. Evaluate R
2. **If R is the worst or 2nd worst**
3. **Evaluate IC or OC**
4. If IC or OC does not improve
5. Evaluate S

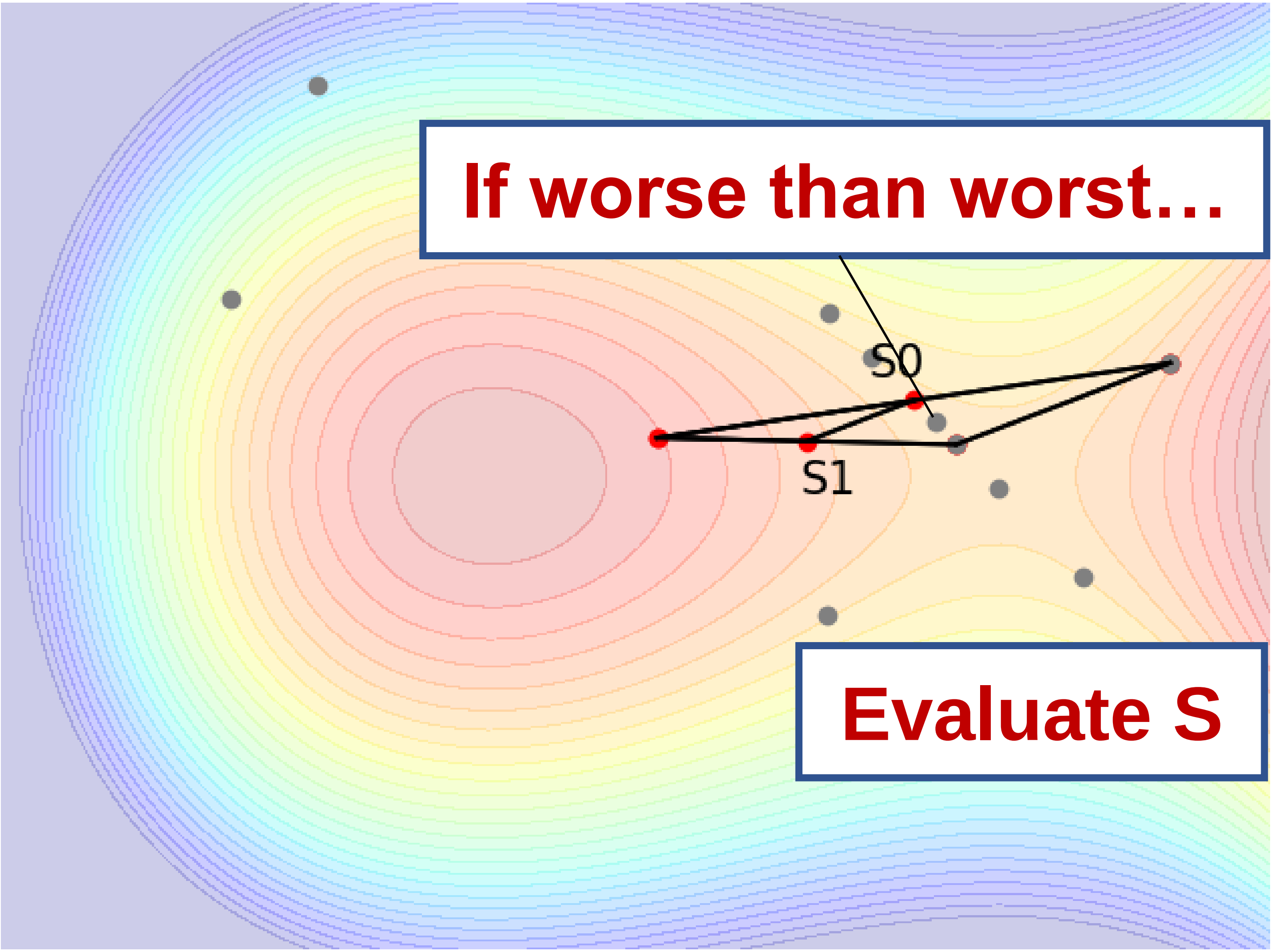


NM Method | Possible Transitions 5, 6 ~part 3~

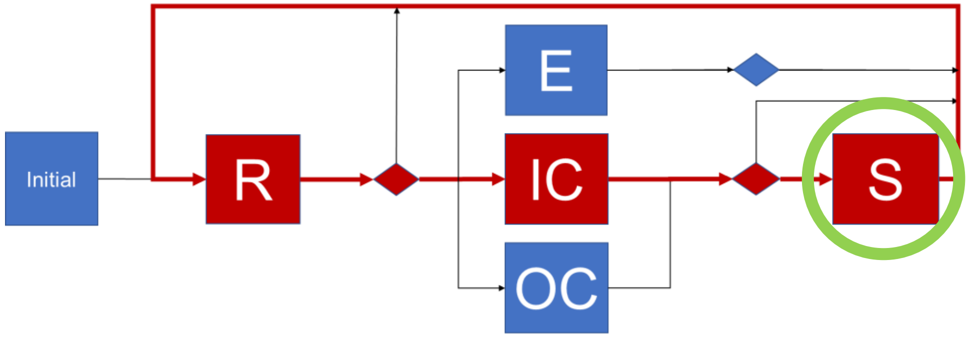


Total: 3 Evaluation Time

1. Evaluate R
2. If R is the worst or 2nd worst
3. Evaluate IC or OC
4. **If IC or OC does not improve**
5. **Evaluate S**

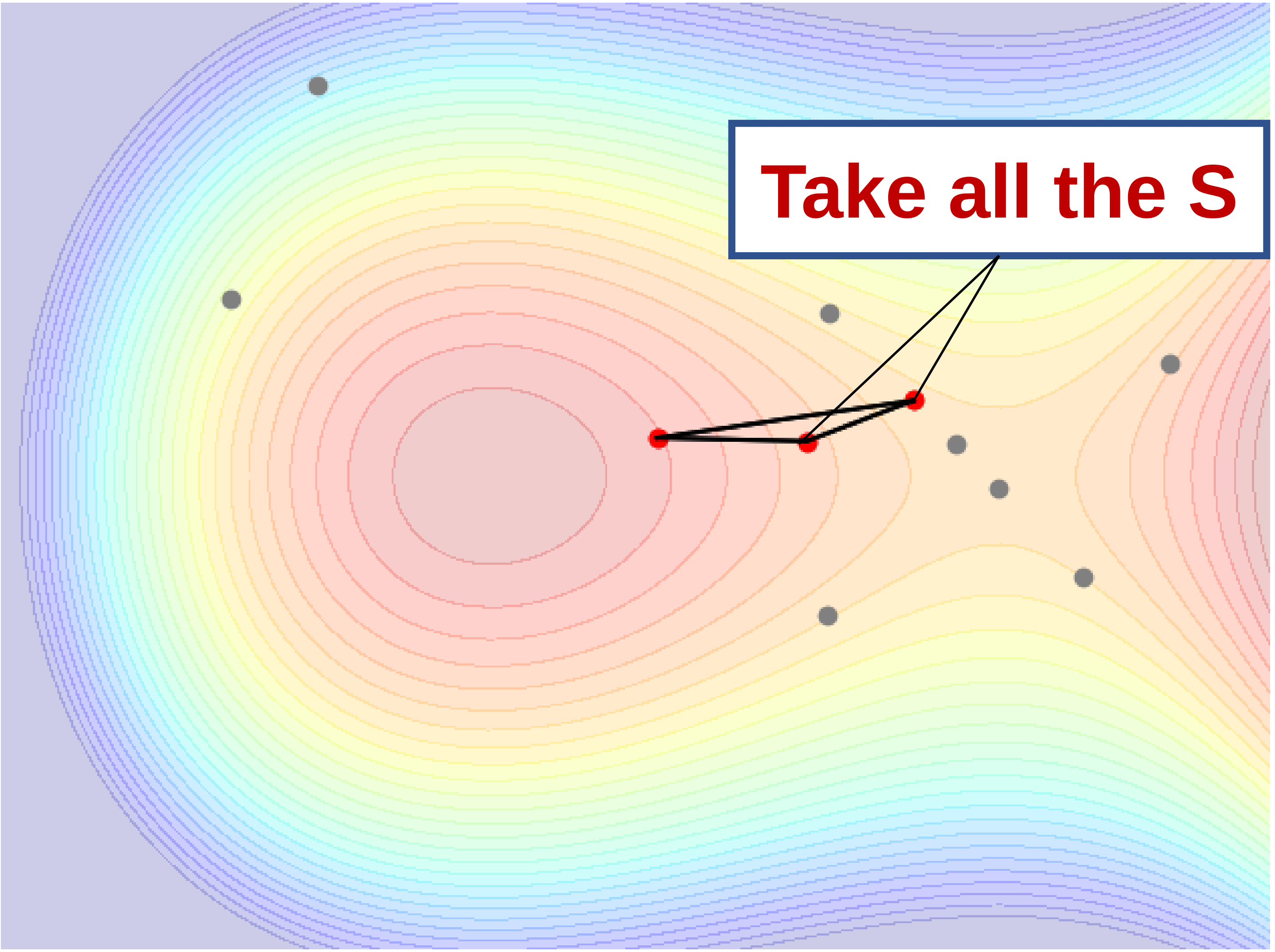


NM Method | Possible Transitions 5, 6 ~part 4~



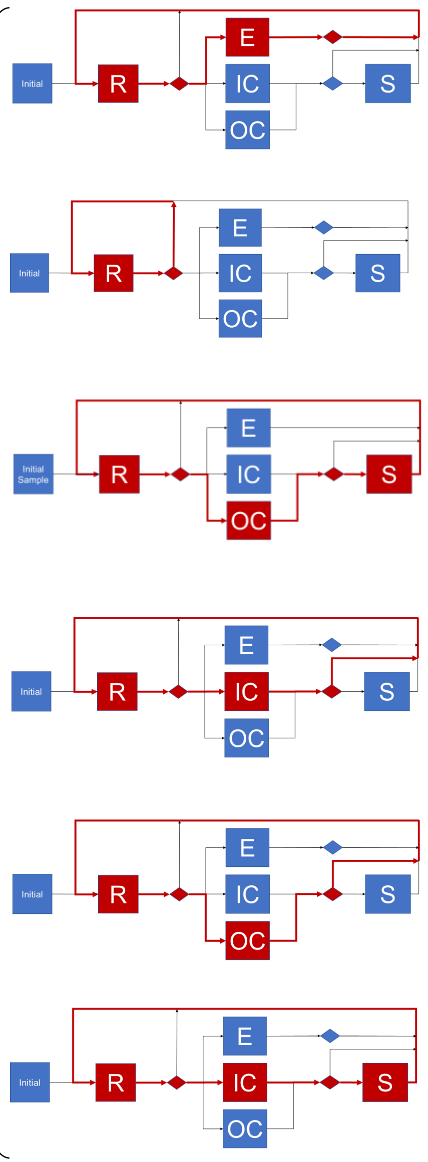
Total: 3 Evaluation Time

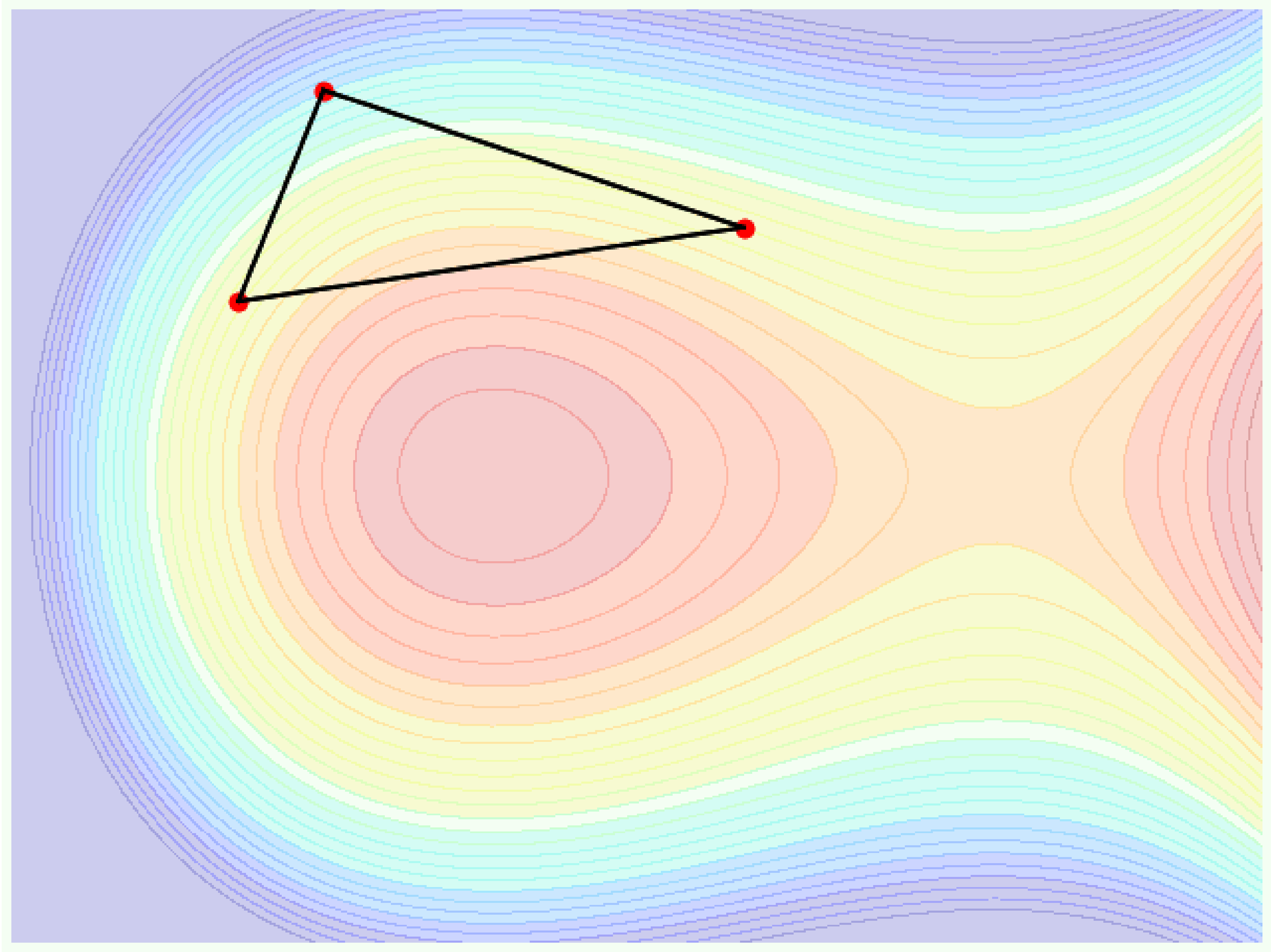
1. Evaluate R
2. If R is the worst or 2nd worst
3. Evaluate IC or OC
4. If IC or OC does not improve
5. **Evaluate S**



Nelder-Mead (NM) Method | Series of Transitions

6 transitions



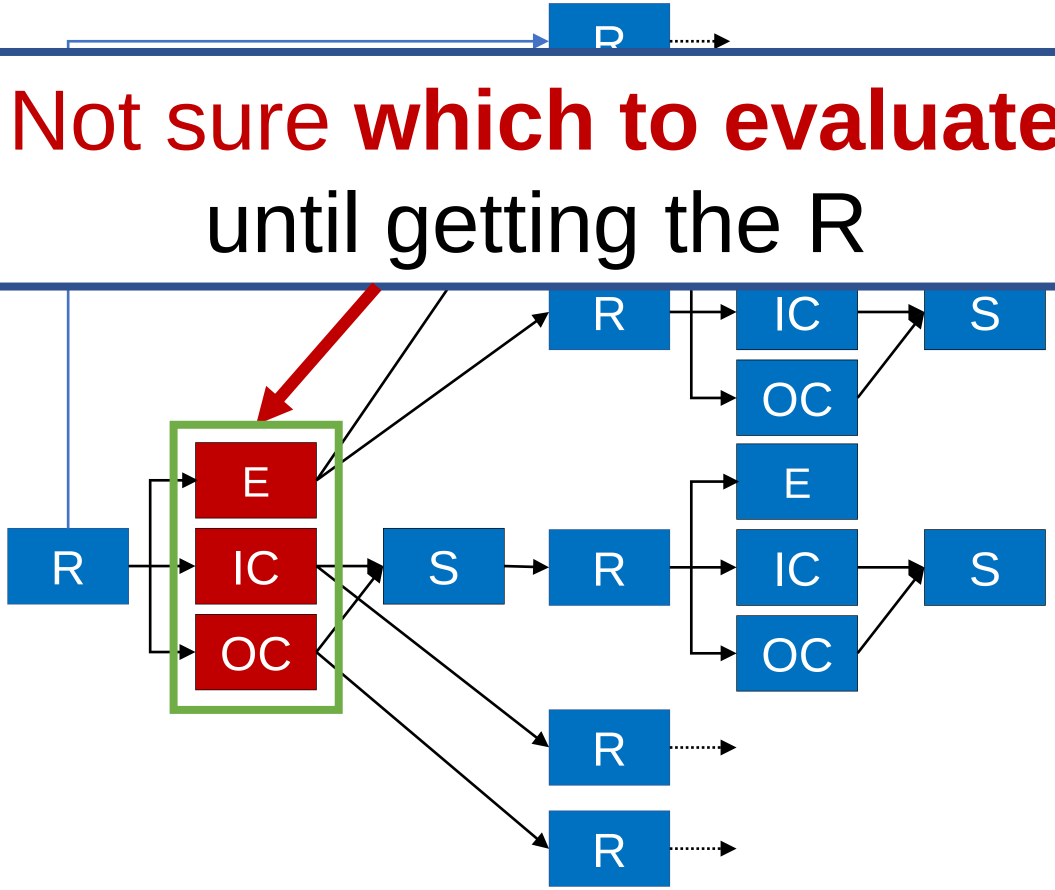


NM Method | The Algorithm Inefficiency

Sequential Method × Expensive Function = # Months

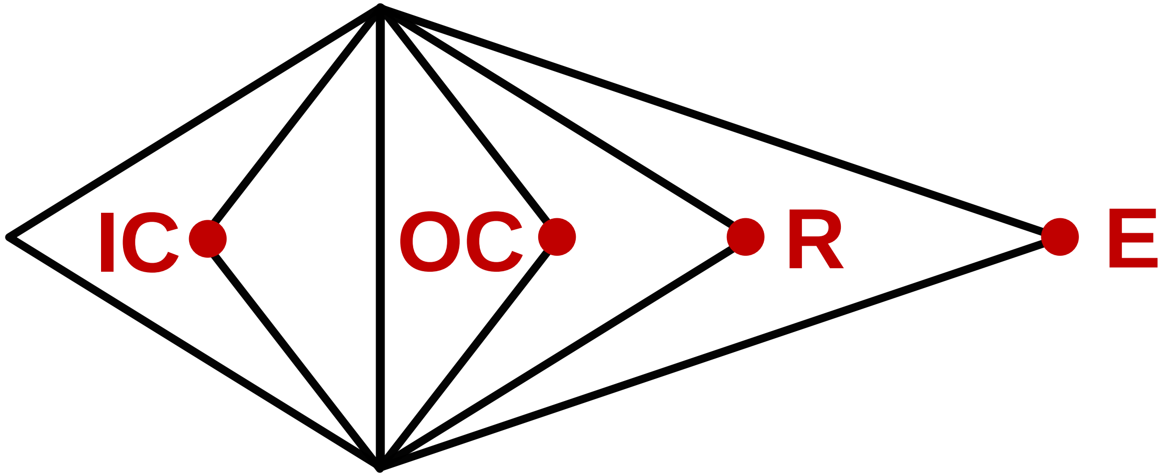
❌ Each evaluation is **expensive** and takes **several hours to days**

✓ However, we can determine the hyperparameter settings' values



Not sure **which to evaluate** until getting the R

However, possible to **Determine the HPs**



Related Work | Naïve Parallel NM Method [Dennis+ 1988]

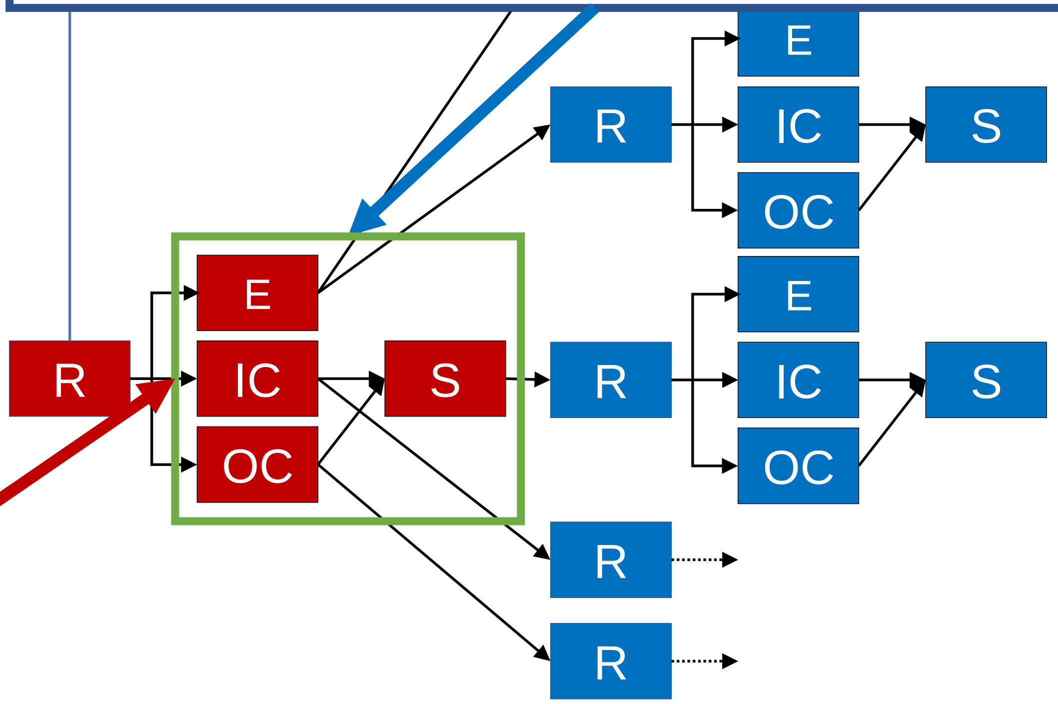
This method computes 1 iteration in parallel

Naïve Parallel NM method

✓ Guarantees to proceed 1 iteration

✗ Evaluate many redundant points

Evaluates all the possible points in parallel



Most of them are not required to proceed the Algorithm

Related Work | Gaussian Process NM Method [Ozaki+ 2019]

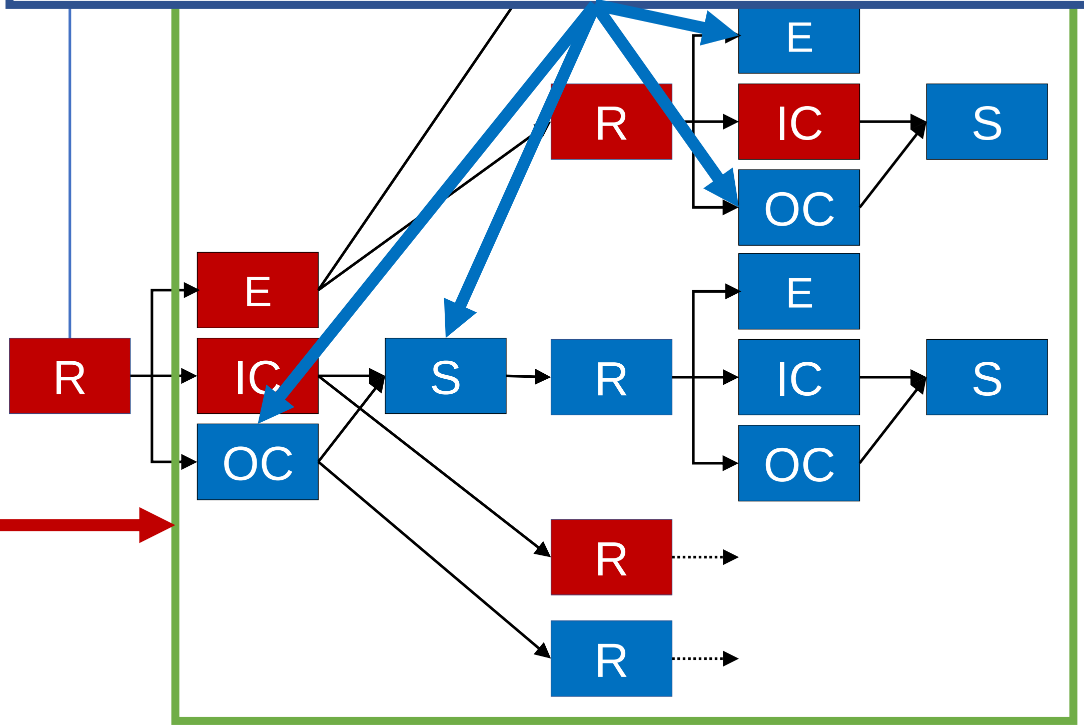
GP approximates the function and gives us next operation

Gaussian Process (**GP**) NM method

✓ Removes redundant points

✗ Not works on noisy high-dim cases

Removes improbable points using GP every epoch



Difficult to predict the exact transition in the case of noisy high-dim function

Solution for Problems of the previous methods

Control of trade-off between completeness and speed

✓ Naïve method **guarantees to proceeds 1 iteration**

✗ However, **inefficient** and **slow**

✓ GP method **removes redundant** points

✗ However, tends to **remove required points** in case of noisy high-dim function

Solution for these problems

✓ **Analyze the behavior** of noisy high-dim function **statistically**

✓ **Use the statistical information** to remove the redundant points

Good Aspects of Proposed Data Collection

Collect the data from benchmark functions *

✓ Can evaluate many kinds of functions and average the statistics

✓ Easy to collect the data of noisy high-dim functions

✓ Not requires substantial amount of time to collect many data

Generalize By diverse functions	**High dimension And Noisy function**	**Inexpensive That's why Many data**

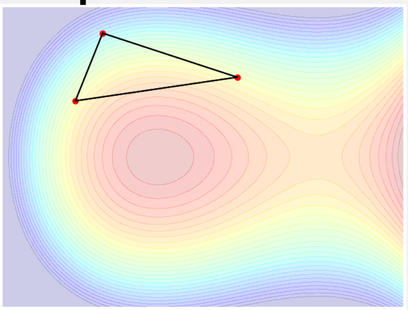
* https://www.sfu.ca/~ssurjano/optimization.html

Settings for Data Collection

Settings for the data collection

1. Optimize the benchmark functions
2. Compute the probabilities of each operation taken in 1 iteration

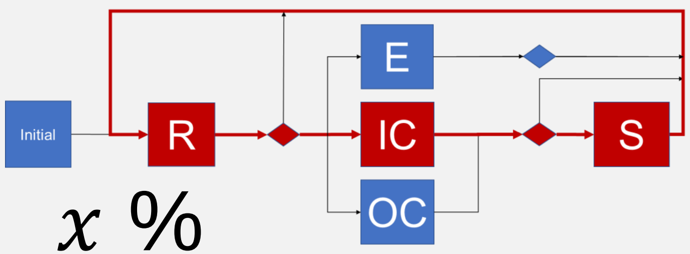
Optimization



× 100 times × 17 functions

Average →

Probability of transition

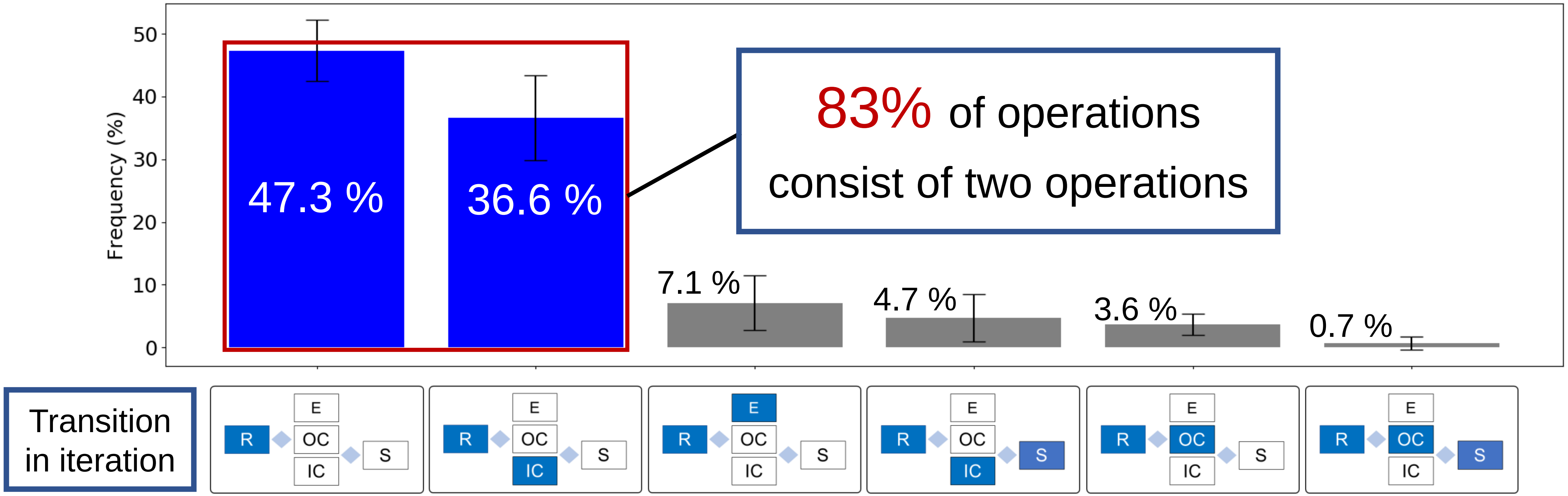


# of dims	10
# of evaluations	100
# of functions	17 *

Statistics of NM Method

Some operations occurs more often than the others

- 2 out of 6 transitions occupy the 83% of transitions in 1 iteration
- It is not reasonable to treat other 4 operations in the same way

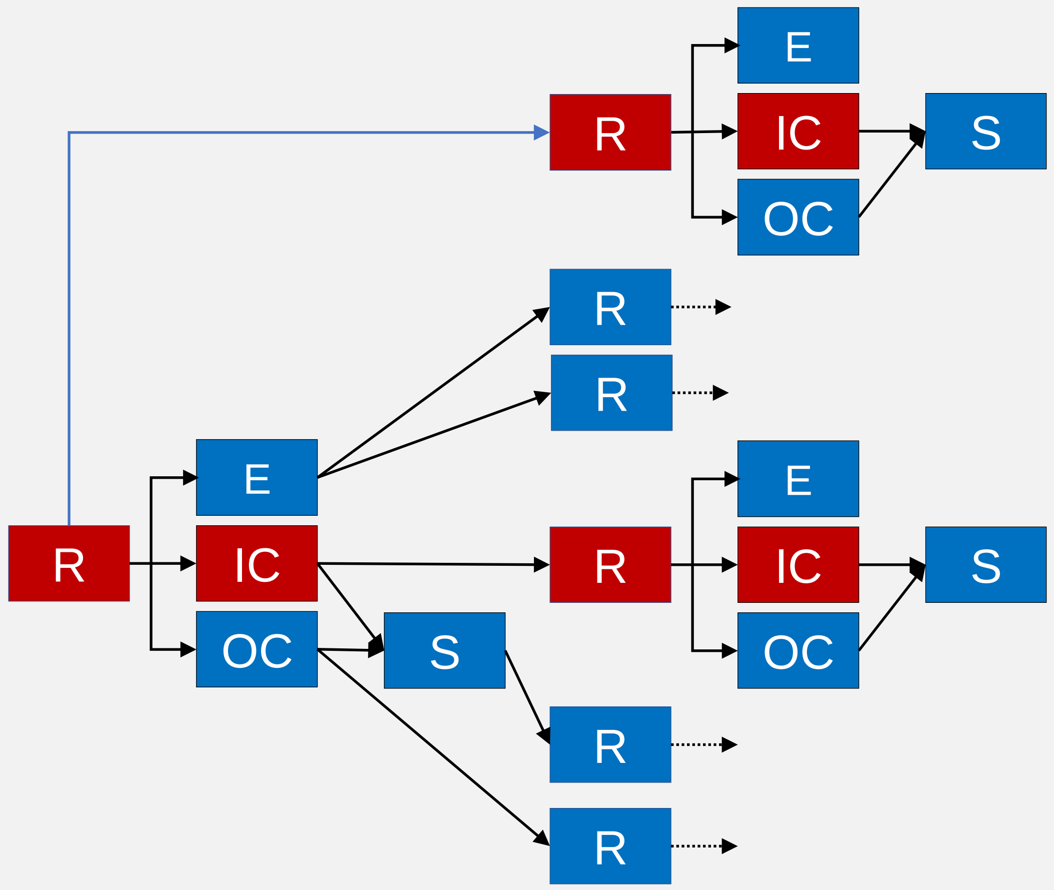


Using probabilities to transition to next operation

- Takes the transitions with high probabilities with a priority

Algorithm

1. Determines points to be evaluated
2. The points will be taken by the order of statistical probabilities
3. Evaluates the points in parallel
4. Iterate 1. to 3. until termination

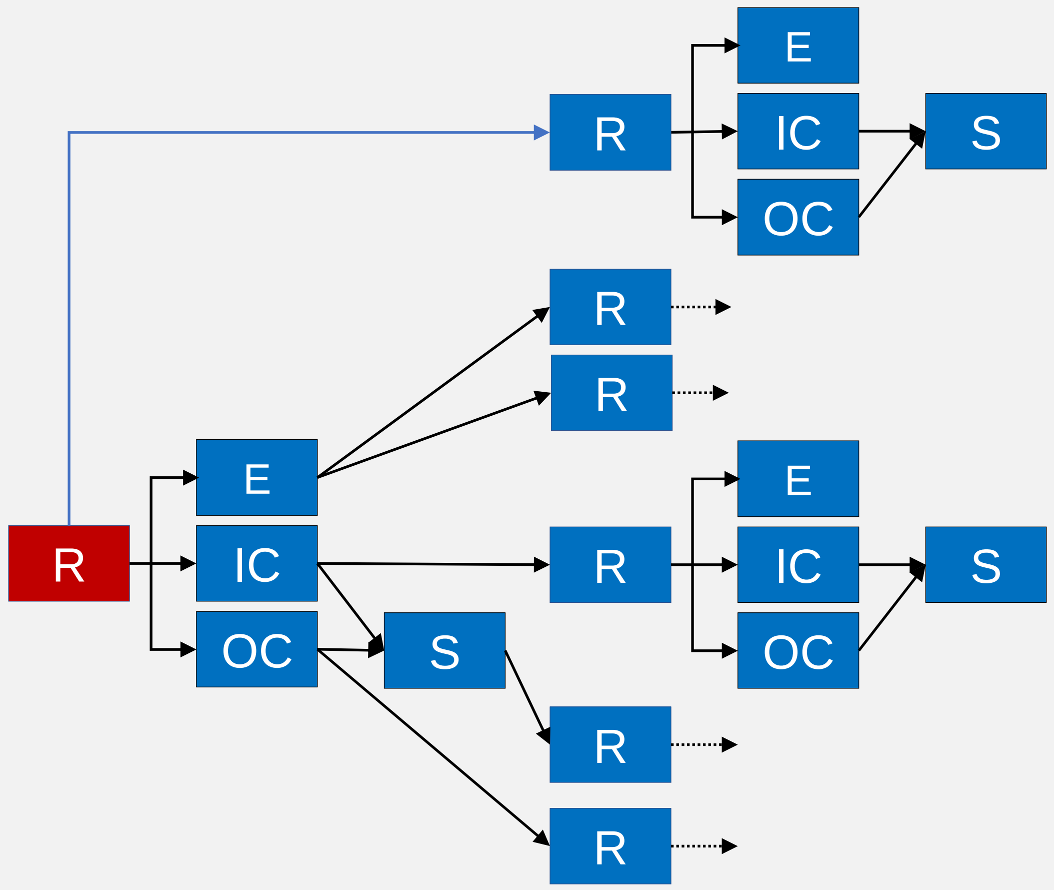


Using probabilities to transition to next operation

- Takes the transitions with high probabilities with a priority

Algorithm

1. Determines points to be evaluated
2. The points will be taken by the order of statistical probabilities
3. Evaluates the points in parallel
4. Iterate 1. to 3. until termination



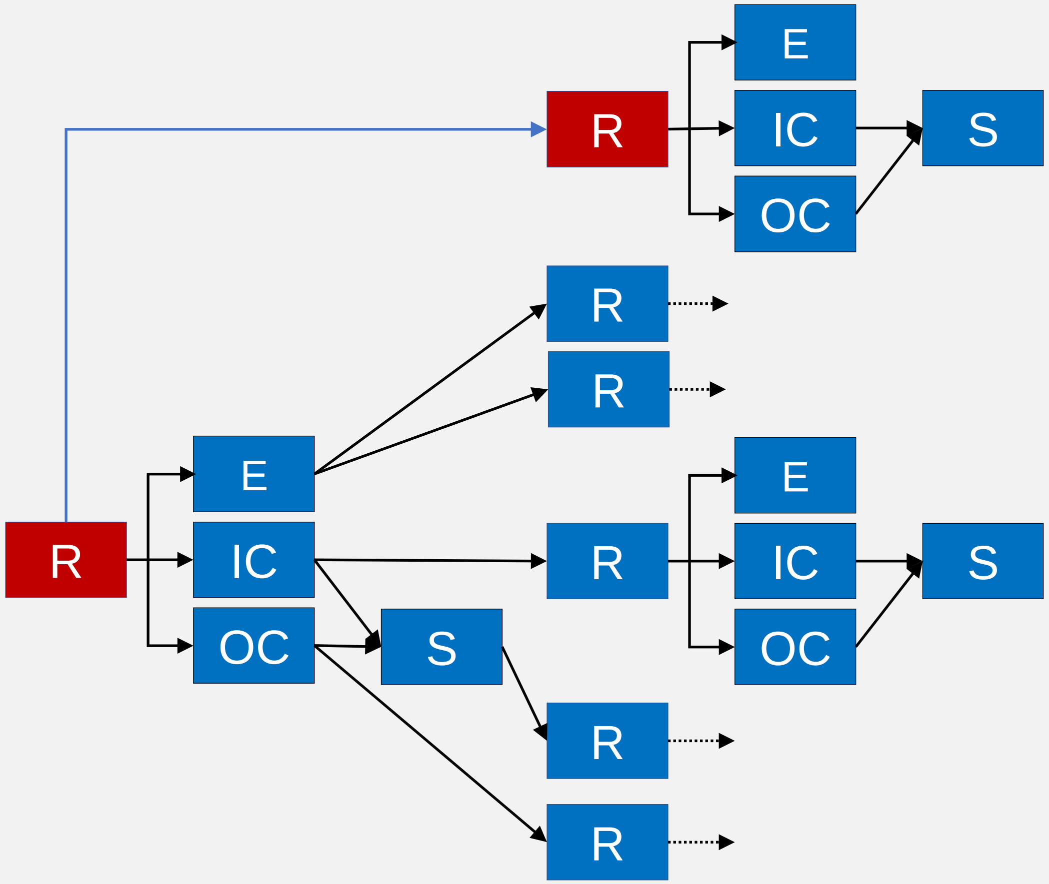
Proposed Method | Example ~ # of processors: 2 ~

Using probabilities to transition to next operation

- Takes the transitions with high probabilities with a priority

Algorithm

1. Determines points to be evaluated
2. The points will be taken by the order of statistical probabilities
3. Evaluates the points in parallel
4. Iterate 1. to 3. until termination



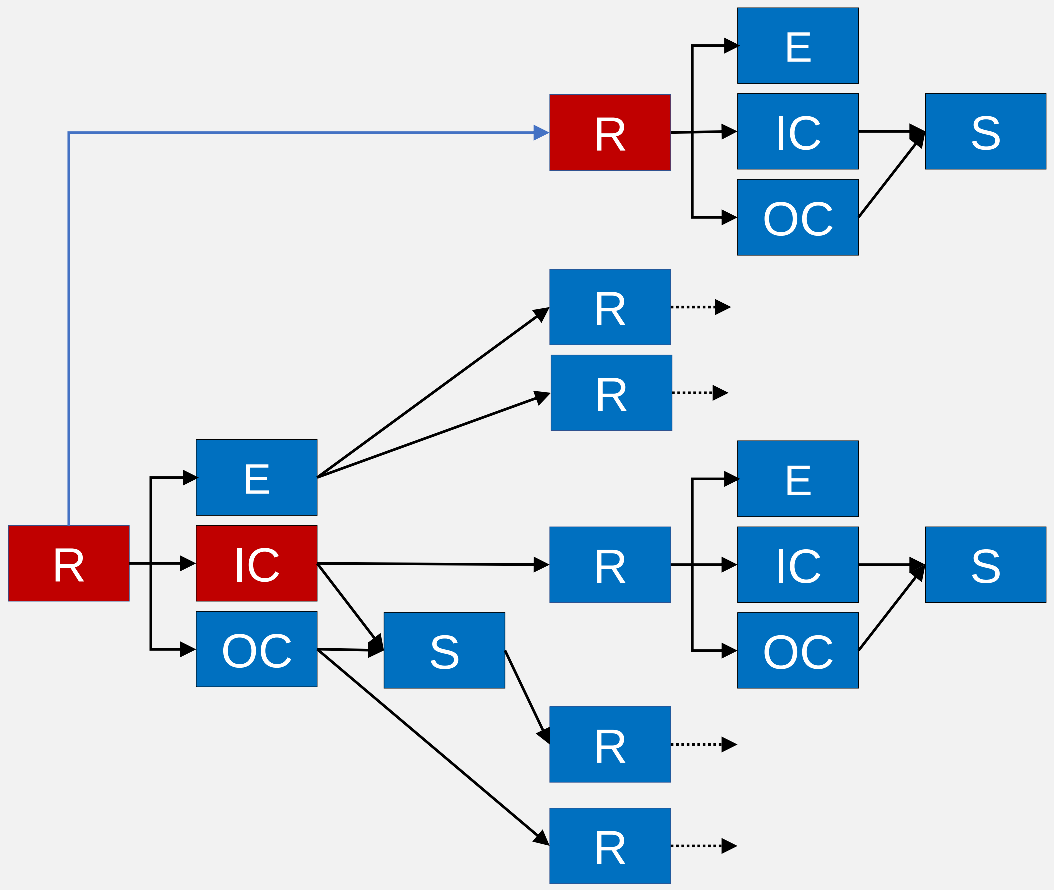
Proposed Method | Example ~ # of processors: 3 ~

Using probabilities to transition to next operation

- Takes the transitions with high probabilities with a priority

Algorithm

1. Determines points to be evaluated
2. The points will be taken by the order of statistical probabilities
3. Evaluates the points in parallel
4. Iterate 1. to 3. until termination

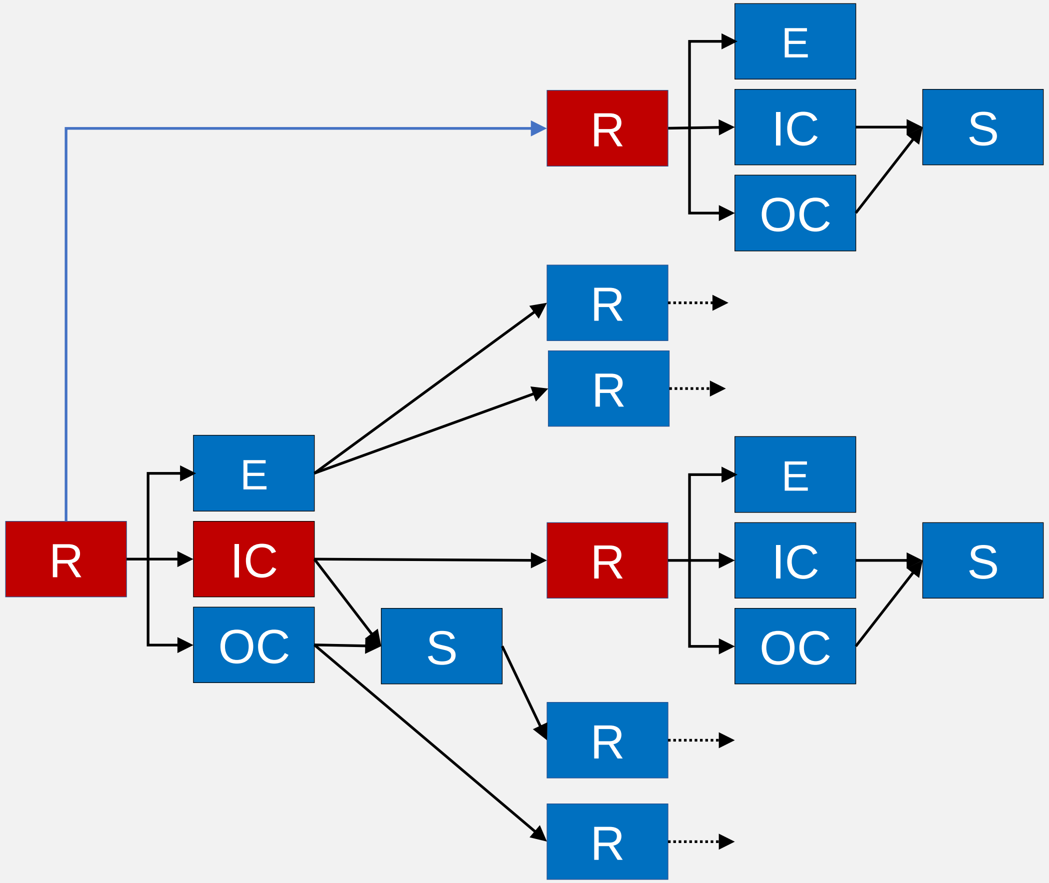


Using probabilities to transition to next operation

- Takes the transitions with high probabilities with a priority

Algorithm

1. Determines points to be evaluated
2. The points will be taken by the order of statistical probabilities
3. Evaluates the points in parallel
4. Iterate 1. to 3. until termination



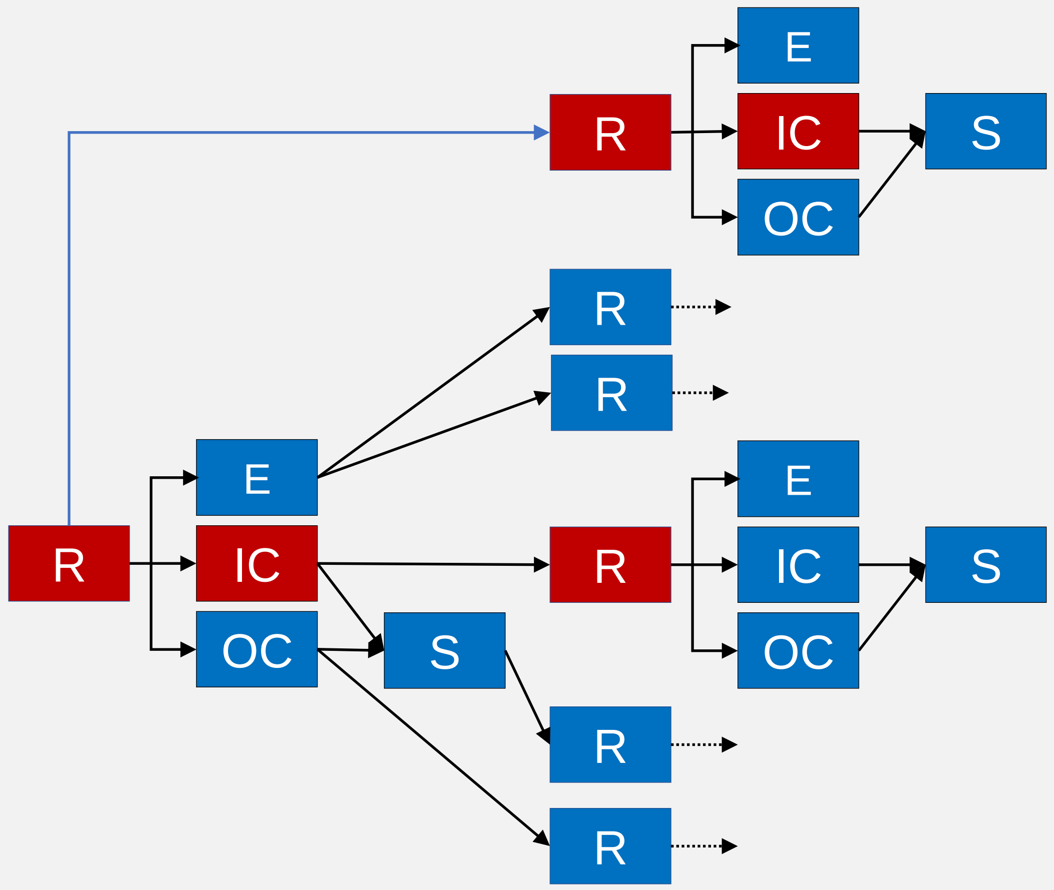
Proposed Method | Example ~ # of processors: 5 ~

Using probabilities to transition to next operation

- Takes the transitions with high probabilities with a priority

Algorithm

1. Determines points to be evaluated
2. The points will be taken by the order of statistical probabilities
3. Evaluates the points in parallel
4. Iterate 1. to 3. until termination



Comparison of Parallel NM Methods | Settings

Comparing NM with the other parallel HPO methods

1. Parallel NM Methods

- Naïve Parallel NM method (NP-NM) [Dennis+ 1988]
- Gaussian Process based NM method (GP-NM) [Ozaki+ 2019]

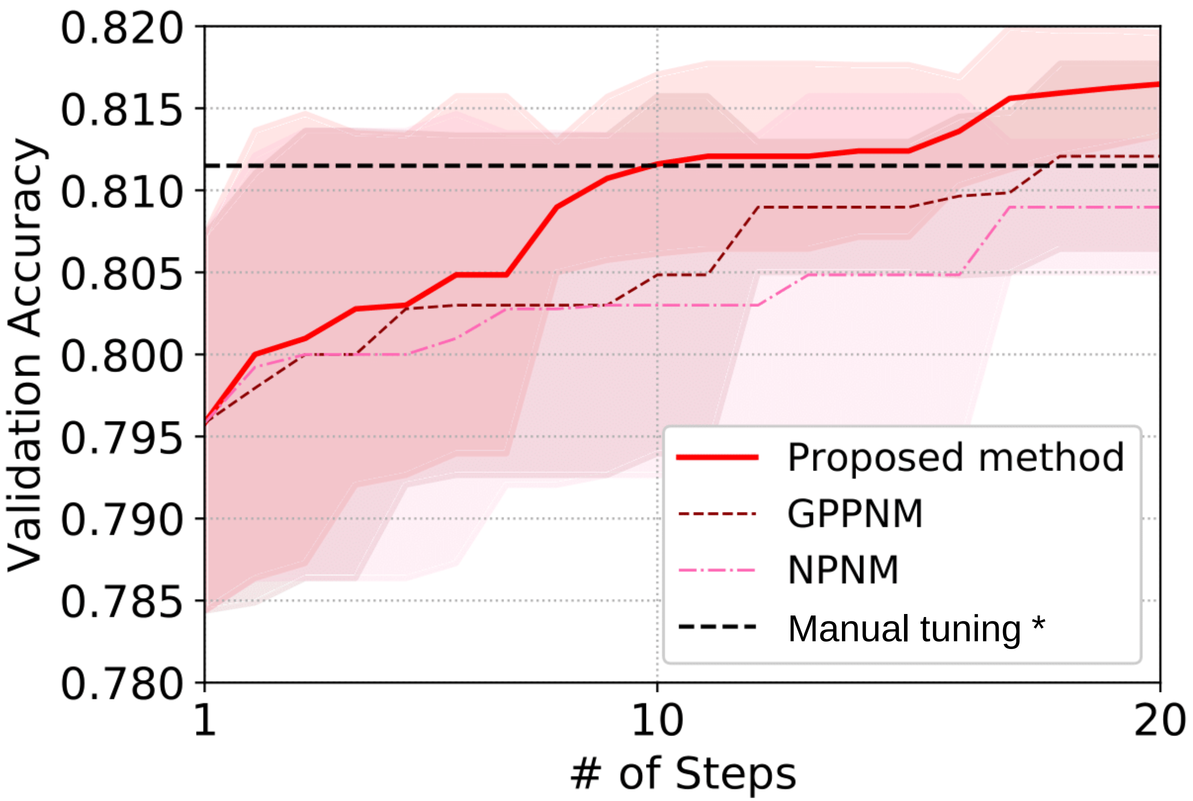
2. The target of the optimization

Classifier	Wide Residual Networks 28-10 [Zagoruyko+ 2016]
# of HPs	11 (Learning rate, Momentum, Weight decay etc…)
# of evaluations	100 (**17 GPUdays**)
# of GPUs	12
Dataset	CIFAR100 (Training 50k, Validation 10k)

Comparison of Parallel NM Methods | Results 1

Achieved better performance than manual tuning [Zagoruyko+ 2016]

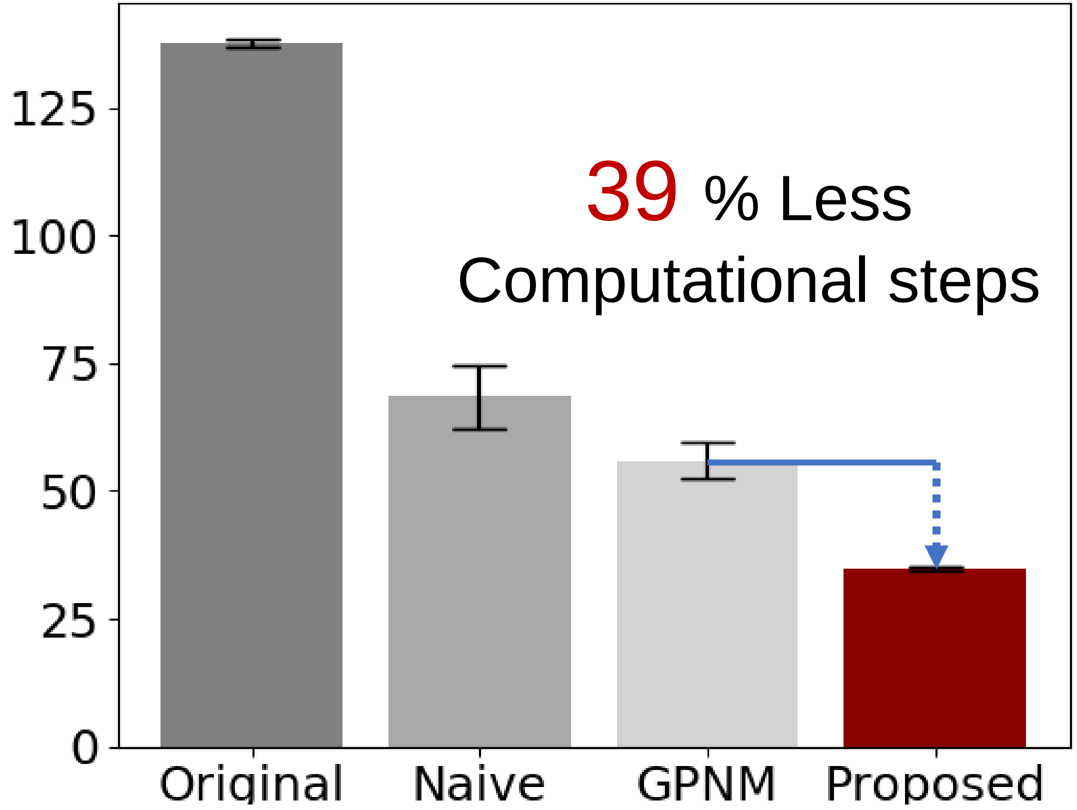
- Found good HP settings **with a limited budget**
- NM method worked successfully on a complex model



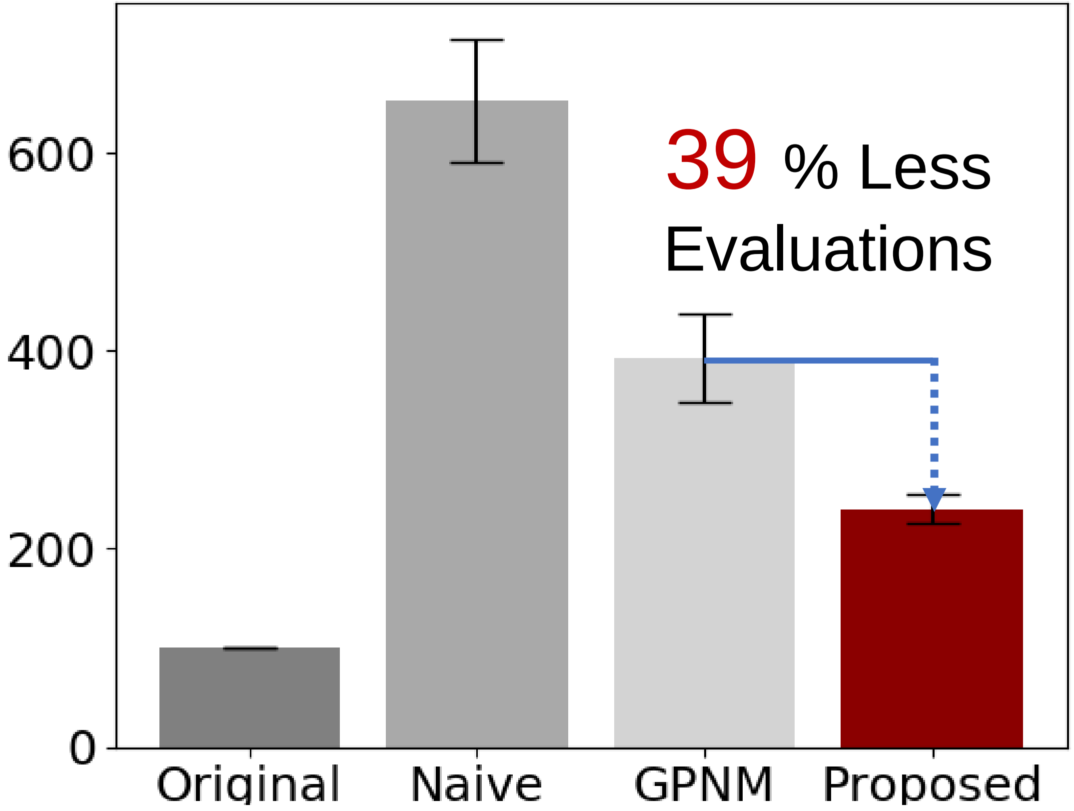
Comparison of Parallel NM Methods | Results 2

Converged only with 39% less steps and evaluations

of steps *



of evaluations



* # of steps = # of evalauations / # of GPUs

Performance Evaluation of Parallel NM | Settings

Comparing NM with the other parallel HPO methods

1. Parallel HPO Methods

- Parallel Bayesian Optimization via Thompson Sampling (BOTS) [Kandasamy+ 2018]
- Random Search (The most naïve parallel approach)

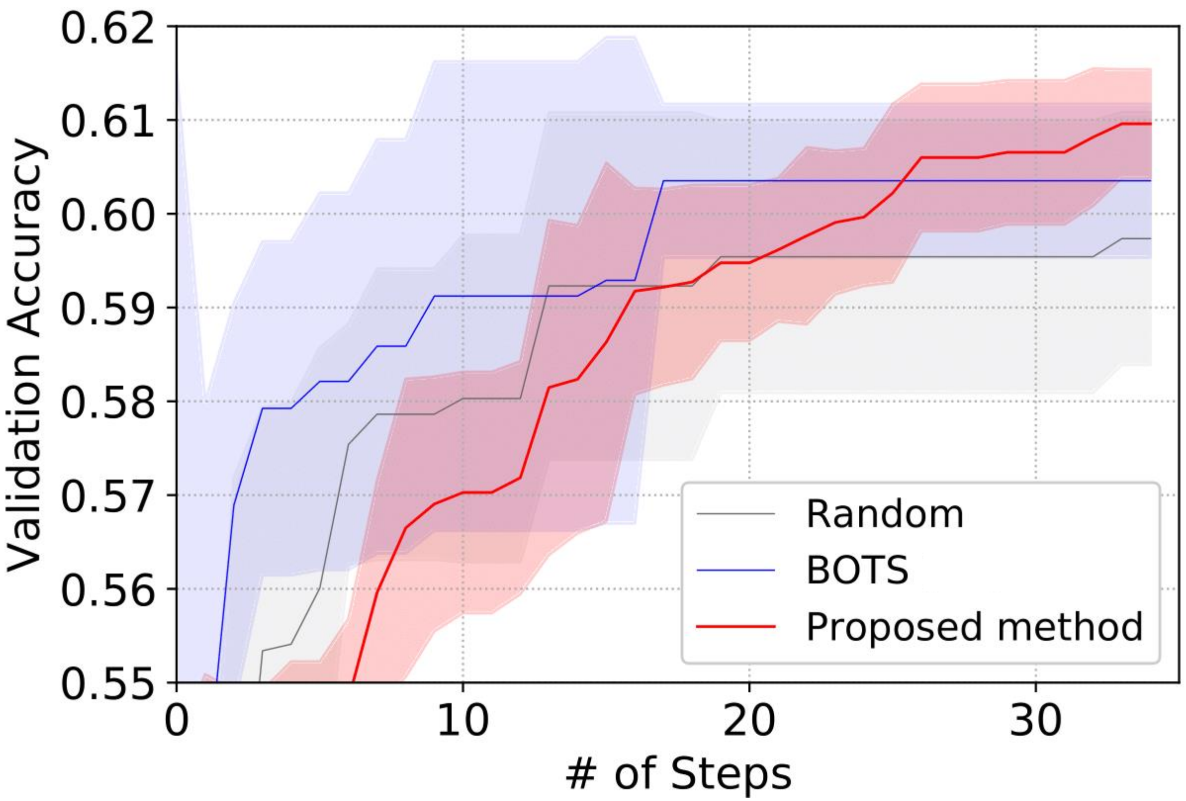
2. The target of the optimization

Classifier	Naïve 8-layer CNN
# of HPs	9 (Learning rate, Momentum, Weight decay etc…)
# of evaluations	150 (**6 GPUdays**)
# of GPUs	4
Dataset	CIFAR100 (Training 50k, Validation 10k)

Performance Evaluation of Parallel NM | Results

Our method found better HPs faster than other methods

- The parallel **NM converges faster** than Parallel Bayesian optimization
- We denote **# of steps = # of evaluatios / # of GPUs**



Conclusion

	Contents
Proposition	Parallel method for NM method based on statistics on 17 types of benchmark functions
Results	・Better than the other parallel HPO methods ・**Reduced 40%** of evaluations and computational time without decreasing the target's performance
Future Study	・Asynchronic parallel method ・Combining with the Gaussian Process

NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)