Rechenberg 1/5 success rule

Operator

Util

Gevol.evolution.evoperator.util

Description

In 1973 Ingo Rechenberg introduced new mutation method for evolutionary strategies: Rechenberg 1/5 success rule. It is based on the opinion that if mutation has improved the individual enough times then we should increase the mutation power to get faster good results.

$$\sigma = \begin{cases} \sigma, P(k) = 1/5 \\ \sigma * \theta, P(k) < 1/5 \\ \frac{\sigma}{\theta}, P(k) > 1/5 \end{cases}$$

Where

- σ - mutation power, how strong the individual will be changed
- θ(k) - parameter how strong the mutation power should be changed, Schwefel in 1981 has found out in his experiment that good value for that parameter is 0,82.
- P(k) - ratio how many successful mutation I_S was performed after k iterations of the total iterations I.

$$P(k) = \frac{I_s}{I}, when\ I > k$$

- 1/5 - ratio set by Rechenberg

In this implementation, the method was adapted for self-adapting operator selector. Instead of mutation power there is probability to apply some operator. The rule may be said, if the operator is able to introduce better individual, his probability of being chosen should be increased.

For each iteration from the set of operators is chosen only one. At the beginning each operator has exactly the same chance to being chosen. Later the probability is changed according to Rechenber 1/5 success rule.

Parameters

1. Objective function - function which is used for evaluation, before and after applying parameter the population is evaluated to get the best score, then it is validated if new has better the best individual or not.
2. Ratio, default ratio is 1/5 (0.2), but it may be changed.
3. Modify power - default value is set to 0.85, but it may be changed.
4. Execution limit - after that number of executions the probability will be investigated according to the Rechenber rule.
5. Operators - list of operators, in each iteration one of them will be applied

Pseudocode

```
TheBestScore = Evaluate(P)
Operator = rand operator(L_Op, L_P)
P' = Operator.Apply(P)
TheBestNewScore = Evaluate(P')
L_Op = RechenbergRule(TheBestScore, TheBestNewScore, Operator, L_Op)
```

Where
P – population
P’ – new population
L_{Op} – List of operators
L_P - List of probabilities to be chosen for each operator

Implementation details

Rand the operator. Each operator may have different probability. All probabilities are stored in the list. So, it rands number between 0 and sum of all probabilities and then it checks in which operator range it is.

```
//find the operator
double probabilitiesSum = 0;
int operatorIndex = -1;
for(int i = 0; i < _operators.Count; i++)
{
	probabilitiesSum += _probabilities[i];
	if(probabilityValue <= probabilitiesSum)
	{
		operatorIndex = i;
		break;
	}
}
```

In fact, each operators has three values in four lists:

1. _operators – list with operators
2. _probabilities – the probability for each operator
3. _succeeded – how many times the operator generated better individual
4. _executions – how many times the operator was triggered

When the operator has been selected, his index is known. Then according to result the Rechenber rule may be applied and it will update necessary lists.

```
_executions[operatorIndex]++;
if(theBestNewScore < theBestScore)
{
	_succeeded[operatorIndex]++;
}
double ratio = _succeeded[operatorIndex] / _executions[operatorIndex];
if (ratio == _ratio || _executions[operatorIndex] < _executionLimit)
	return;
if (ratio < _ratio)
{
	//decrease probability
	_probabilities[operatorIndex] *= _modifyPower;
}
else
{
	//increase probability
	_probabilities[operatorIndex] /= _modifyPower;
}
```

References

1. “The theory of evolution strategies”, Hans-Georg Beyer, Springer 2001
2. "Parallel problem solving from nature--PPSN IV: International Conference on Evolutionary Computation, the 4th International Conference on Parallel Problem

Solving from Nature, Berlin, Germany, September 22-26, 1996 : proceedings", Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, Springer, 1996