2015 Third International Conference on Artificial Intelligence, Modelling and Simulation

BSKF: Binary Simulated Kalman Filter

Zulkifli Md Yusof, Ismail Ibrahim,
Siti Nurzulaikha Satiman, and Zuwairie Ibrahim
Faculty of Electrical and Electronics Engineering
Universiti Malaysia Pahang
26600 Pekan, Pahang, Malaysia

Nor Hidayati Abd Aziz and Nor Azlina Ab Aziz
Faculty of Engineering and Technology
Multimedia University
75450 Melaka, Malaysia

***Abstract*—Inspired by the estimation capability of Kalman filter, we have recently introduced a novel estimation-based optimization algorithm called simulated Kalman filter (SKF). Every agent in SKF is regarded as a Kalman filter. Based on the mechanism of Kalman filtering and measurement process, every agent estimates the global minimum/maximum. Measurement, which is required in Kalman filtering, is mathematically modelled and simulated. Agents communicate among them to update and improve the solution during the search process. However, the SKF is only capable to solve continuous numerical optimization problem. In order to solve combinatorial optimization problems, an extended version of SKF algorithm, which is termed as Binary SKF (BSKF), is proposed. Similar to existing approach, a mapping function is used to enable the SKF algorithm to operate in binary search space. A set of traveling salesman problems are used to evaluate the performance of the proposed BSKF against Binary Gravitational Search Algorithm (BGSA) and Binary Particle Swarm Optimization (BPSO).**

Keywords-simulated kalman filter; traveling salesman problem; combinatorial optimization

I. INTRODUCTION

There are various meta-heuristic algorithms exist in literature nowadays. However, not all meta-heuristic algorithms were originally developed to operate in binary search space. An example of this algorithm is simulated Kalman filter (SKF), which has been recently introduced by Ibrahim *et al.* in 2015 [1]. The SKF is a population-based optimization algorithm that is inspired by the estimation capability of Kalman filter. In order to solve discrete optimization problems with SKF, modification or enhancement is needed. An example of such modification is by using a sigmoid function as a mapping function to let particle swarm optimization (PSO) to operate in binary search space [2].

The objective of this study is to extend the SKF algorithm for solving combinatorial optimization problem. Similar to existing works, a mapping function is employed to enable the SKF algorithm to operate in binary search space.

This paper is organized as follows. At first, SKF will be briefly reviewed followed by a detail description of the proposed Binary SKF (BSKF) algorithm. Experimental set up will be explained, results will be shown and discussed. Lastly, a conclusion will be provided at the end of this paper.

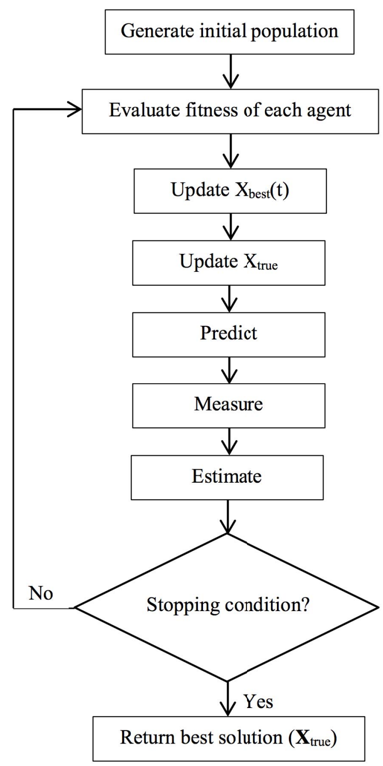

Figure 1. The original simulated Kalman filter (SKF) algorithm.

II. SIMULATED KALMAN FILTER

Every agent in SKF is regarded as a Kalman filter. Based on the mechanism of Kalman filtering and measurement process, every agent estimates the global minimum/maximum. Measurement, which is required in Kalman filtering, is mathematically modelled and simulated. Agents communicate among them to update and improve the solution during the search process. The simulated Kalman filter (SKF) algorithm is illustrated in Figure 1.

Consider n number of agents, SKF algorithm begins with initialization of n agents, in which the states of each agent are given randomly. The maximum number of iterations, t_{max}, is defined. The initial value of error covariance estimate, $P(0)$, the process noise value, Q, and the measurement noise value, R, which are required in Kalman filtering, are also defined during initialization stage.

978-1-4673-8675-3/15 $31.00 © 2015 IEEE
DOI 10.1109/AIMS.2015.23

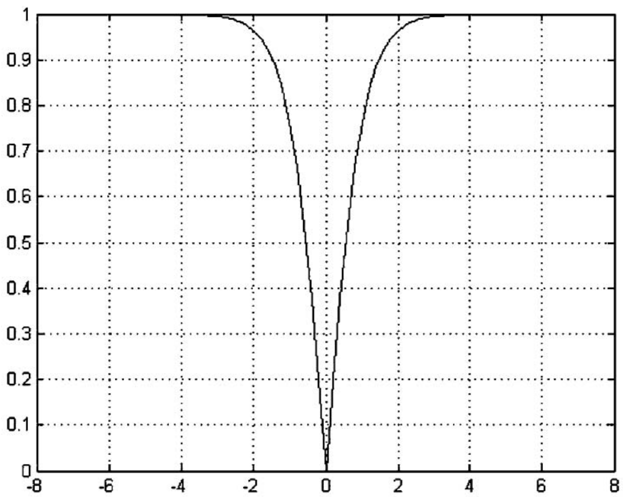

Figure 2. A mapping function used in [3].

Then, every agent is subjected to fitness evaluation to produce initial solutions $\{\boldsymbol{X}_1(0), \boldsymbol{X}_2(0), \boldsymbol{X}_3(0), \ldots, \boldsymbol{X}_{n-2}(0), \boldsymbol{X}_{n-1}(0), \boldsymbol{X}_n(0)\}$. The fitness values are compared and the agent having the best fitness value at every iteration, t, is registered as $\boldsymbol{X}_{\mathbf{best}}(t)$. For function minimization problem,

$$\boldsymbol{X}_{\mathbf{best}}(t) = \min_{i \in 1,\ldots,n} fit_i(\boldsymbol{X}(t)) \quad (1)$$

whereas, for function maximization problem,

$$\boldsymbol{X}_{\mathbf{best}}(t) = \max_{i \in 1,\ldots,n} fit_i(\boldsymbol{X}(t)) \quad (2)$$

The-best-so-far solution in SKF is named as $\boldsymbol{X}_{\mathbf{true}}$. The $\boldsymbol{X}_{\mathbf{true}}$ is updated only if the $\boldsymbol{X}_{\mathbf{best}}(t)$ is better than the $\boldsymbol{X}_{\mathbf{true}}$. ($\boldsymbol{X}_{\mathbf{best}}(\boldsymbol{t}) < \boldsymbol{X}_{\mathbf{true}}$ for minimization problem, or $\boldsymbol{X}_{\mathbf{best}}(\boldsymbol{t}) > \boldsymbol{X}_{\mathbf{true}}$ for maximization problem).

The subsequent calculations are largely similar to the predict-measure-estimate steps in Kalman filter. In the prediction step, the following time-update equations are computed.

$$\boldsymbol{X}_i(\boldsymbol{t}|\boldsymbol{t}) = \boldsymbol{X}_i(\boldsymbol{t}) \quad (3)$$

$$P(t|t) = P(t) + Q \quad (4)$$

where $\boldsymbol{X}_i(t)$ and $\boldsymbol{X}_i(t|t)$ are the current state and current transition/predicted state, respectively, and $P(t)$ and $P(t|t)$ are the current error covariant estimate and current transition error covariant estimate, respectively. Note that the error covariant estimate is influenced by the process noise, Q.

The next step is measurement, which is a feedback to estimation process. Measurement is modeled such that its output may take any value from the predicted state estimate, $\boldsymbol{X}_i(\boldsymbol{t}|\boldsymbol{t})$, to the true value, $\boldsymbol{X}_{\mathbf{true}}$. Measurement, $\boldsymbol{Z}_i(t)$, of each individual agent is simulated based on the following equation:

$$\boldsymbol{Z}_i(\boldsymbol{t}) = \boldsymbol{X}_i(\boldsymbol{t}|\boldsymbol{t}) + \sin(rand \times 2\pi) \times |\, \boldsymbol{X}_i(\boldsymbol{t}|\boldsymbol{t}) - \boldsymbol{X}_{\mathbf{true}}| \quad (5)$$

The $\sin(rand \times 2\pi)$ term provides the stochastic aspect of SKF algorithm and $rand$ is a uniformly distributed random number in the range of [0,1].

The final step is the estimation. During this step, Kalman gain, $K(t)$, is computed as follows:

$$K(t) = P(t|t) \,/\, (P(t|t)+R) \quad (6)$$

Then, the estimation of next state, $\boldsymbol{X}_i(t+1)$, and the updated error covariant are computed based on Eqn. (7) and Eqn. (8), respectively.

$$\boldsymbol{X}_i(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{X}_i(\boldsymbol{t}|\boldsymbol{t}) + \boldsymbol{\Delta}_i \quad (7)$$

$$P(t+1) = (1 - K(t)) \times P(t|t) \quad (8)$$

where $\boldsymbol{\Delta}_i = K(t) \times (\boldsymbol{Z}_i(\boldsymbol{t}) - \boldsymbol{X}_i(\boldsymbol{t}|\boldsymbol{t}))$. Finally, the next iteration is executed until the maximum number of iterations, $t_{\max}$, is reached.

III. Binary Simulated Kalman Filter

In order to solve a combinatorial optimization problem using SKF, the $\boldsymbol{\Delta}_i$ term in Eqn. (7) is mapped into a probabilistic value [0,1]. Then, the probabilistic value is compared to a random number [0,1] to update a bit string. In BSKF, most calculations are similar to the original SKF. Modifications are needed only during initialization and generation of solution to combinatorial optimization problem.

During the initialization of agents, a random bit string, Σ_i , is generated for each agent. Each bit in the bit string is associated to a dimension. The length of the bit string is problem dependent and subjected to the size of the problem.

In binary gravitational search algorithm (BGSA) [3], a function shown in Figure 2 is used to map a velocity value into a probabilistic value within interval [0,1]. Similar function is used in this study. The term Δ_i is mapped to a probabilistic value within interval [0,1] using a mapping function, $S(\boldsymbol{\Delta}_i(t))$, as follows:

$$S(\boldsymbol{\Delta}_i(t)) = |\tanh \boldsymbol{\Delta}_i(t)| \quad (9)$$

After the $S(\boldsymbol{\Delta}_i(t))$ is calculated, a random number, $rand$, is generated and a binary value at dimension d of an ith agent, Σ_i^d, is updated according to the following rule:

if $rand < S(\boldsymbol{\Delta}_i(t))$
    then $\Sigma_i^d(t+1) = \text{complement } \Sigma_i^d(t+1)$
    else $\Sigma_i^d(t+1) = \Sigma_i^d(t+1)$
end (10)

IV. Experiment

The BSKF is applied to solve a set of TSP. The objective of TSP is to find the shortest distance from a start city to an end city while visiting every city not more than once. In this paper, 50 instances of TSPs with various problem sizes were considered, as shown in Table 1. These problems were taken from TSPLib [4].

Experimental setting for BSKF is shown in Table 2. For benchmarking purpose, binary particle swarm optimization (BPSO) [2] and BGSA [3] are also considered. Experimental setting for BPSO and BGSA are shown in Table 3 and Table 4, respectively. As tabulated in Table 5, in all experiments, the number of runs, the number of agents, and the number of iterations are 50, 30, and 1000, respectively.

TABLE I. PROPERTY OF THE TEST PROBLEMS

TSP Index	Name	Size
1	Berlin52	52
2	Bier127	127
3	Ch130	130
4	Ch150	150
5	D198	198
6	D493	493
7	D657	657
8	D1291	1291
9	D2103	2103
10	DSJ1000	1000
11	Eil51	51
12	Eil76	76
13	Eil101	101
14	FL1400	1400
15	FL1577	1577
16	GIL262	262
17	KROA100	100
18	KROA150	150
19	KROA200	200
20	KROB100	100
21	KROB200	200
22	KROC100	100
23	KROD100	100
24	KROE100	100
25	LIN105	105
26	LIN318	318
27	P654	654
28	PCB442	442
29	PR76	76
30	PR107	107
31	PR124	124
32	PR136	136
33	PR144	144
34	PR152	152
35	PR226	226
36	PR264	264
37	PR299	299
38	PR439	439
39	PR1002	1002
40	PR2392	2392
41	RAT99	99
42	RAT195	195
43	RAT575	575
44	RAT783	783
45	RD100	100
46	RL1304	1304
47	RL1323	1323
48	RL1889	1889
49	ST70	70
50	TS225	225

TABLE II. BSKF PARAMETERS

Parameter	Value
Error covariance, *P*	1000
Process noise, *Q*	0.5
Measurement noise, *R*	0.5
rand	[0,1]

TABLE III. BPSO PARAMETERS

Parameter	Value
Inertia weight, ω	0.9-0.4
Cognitive coefficient, c_1	2
Social coefficient, c_2	2

TABLE IV. BGSA PARAMETERS

Parameter	Value
β	20
Initial gravitational value, *Go*	100

TABLE V. EXPERIMENTAL PARAMETERS

Parameter	Value
Number of agent	30
Number of iteration	1000
Number of run	50

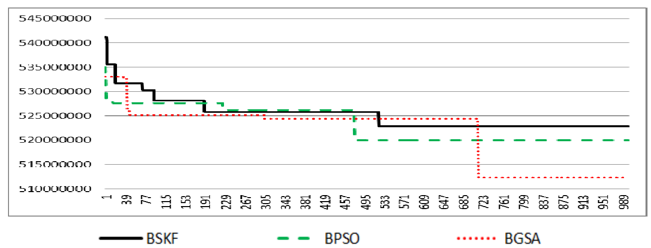

Figure 3. Example of convergence curve for TSP index 10.

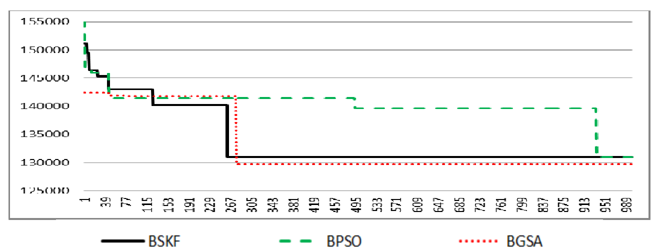

Figure 4. Example of convergence curve for TSP index 24.

V. RESULT AND DISCUSSION

The proposed BSKF is compared with BGSA and BPSO. The average performances and average rankings of the three algorithms are presented in Table 6, Table 7, and Table 8. Based on average performances, Wilcoxon signed rank test is performed and the result are tabulated in Table 9 and Table 10. The level of significant chosen here is $\sigma = 0.05$. It is found that BPSO is ranked first, followed by BGSA and BSKF. Also, BPSO is statistically on par with BGSA. However, statistically, BPSO and BGSA are found to perform significantly better than BSKF in solving the TSP benchmark problems used in this study. Examples of convergence curves are shown in Figure 3, Figure 4, and Figure 5.

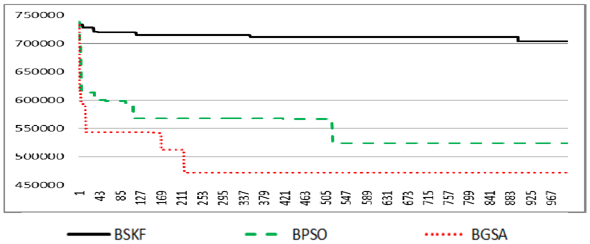

Figure 5. Example of convergence curve for TSP index 28.

TABLE VI. RESULTS (TSP INDEX 1 TO 19)

TSP Index	BSKF	BGSA	BPSO
1	22847.63773	23086.34	22645.1
	±613.801395	±658.5154	±600.0265
2	542440	510137.9	534565.08
	±7797.500078	±10528.6018	±12835.19
3	39267.0007	39533.3	39113.46
	±540.71443	±591.3782	±467.7691
4	46174.03239	46660.14	46159.44
	±549.7645371	±592.7121	±539.7974
5	158476.8621	73367.02	113701.94
	±2628.158368	±5483.8943	±4104.25
6	411621.0081	281229.78	314509.44
	±2981.592683	±50949.6277	±41704.08
7	796929.4	360067	488599.38
	±3727.27173	±95554.9651	±97920.66
8	1648226.699	462130.38	371718.84
	±4690.592383	±327736.2017	±297787.9
9	3123722.879	1521030.08	470854.32
	±11753.86476	±566990.8978	±148393.1
10	523661043.4	543116116	523506056
	±2674818.056	0	±1820245
11	2127.613034	1274.02	1265.18
	±4223.850194	±26.5549	±21.08553
12	23782.28052	2059.6	2036.86
	±107600.0922	±40.1385	±36.22267
13	2853.753981	2694.38	2834.8
	±43.00394958	±81.693	±53.45568
14	1581580.539	218859.2	636912.08
	±6306.738932	±255773.5343	±307582.5
15	1295394.951	457890.44	742191.28
	±4068.37454	±66089.346	±101450
16	23853.89821	23897.04	23829.98
	±213.7886322	±164.5367	±189.9081
17	137188.7233	137849.86	136400.46
	±1627.440808	±2700.4702	±2131.348
18	215796.8988	217679.18	214231.48
	±2757.859867	±2631.9108	±3855
19	291063.7575	293208.18	291490.08
	±3266.748441	±3039.3297	±3357.707

TABLE VII. RESULTS (TSP INDEX 20 TO 43)

TSP Index	BSKF	BGSA	BPSO
20	134786.5094	136383.42	134948.86
	±2062.810956	±2628.9412	±2010.249
21	286095.51788	287752.36	286063.44
	±3287.535707	±3295.4881	±3324.386
22	135539.3112	136650.52	134922.54
	±2411.877566	±2754.4967	±2837.615
23	131396.7992	132814.44	131014.66
	±2160.69733	±1932.453	±2428.343
24	138610.6636	139215.18	137299.96
	±2433.978931	±2506.1192	±2664.33
25	99045.12665	74610.86	92507.64
	±1871.495614	±3058.9377	±3493.191
26	529112.6636	307040.72	316849.4
	±3883.023789	±8386.1147	±31766.29
27	1849636.903	655530.66	746735.08
	±10194.3747	±326570.5102	±293075.8
28	708016.8516	499374.42	547372.76
	±5095.425909	±12363.1119	±48036.28
29	461949.6614	344443.98	440706.06
	±6632.216622	±20144.31771	±31226.07343
30	449263.3366	291552.78	403180.86
	±8949.619695	±19729.08327	±26633.22771
31	579691.2334	439951.78	496858.62
	±9491.056314	±16126.47684	±33503.8591
32	689880.3946	594103.42	569120.42
	±10383.53084	±15941.32192	±54473.13338
33	682410.7628	527286.02	438790.2
	±10987.78483	±18519.56707	±27513.31193
34	886457.25178	527286.02	598525.64
	±13349.23911	±18519.56707	±26413.19748
35	1482490.132	877046.58	1040259.36
	±18816.01491	±42722.34301	±36403.1764
36	958776.7695	594693.06	227510.52
	±9420.712553	±31582.01513	±31700.3675
37	666494.5932	357120.62	348218.58
	±5973.1801353	±73865.05996	±16557.74074
38	1731522.587	980018.4	1234072.08
	±16724.59614	±207452.2873	±141913.8577
39	6079543.192	1698504.32	2031444.24
	±32844.04673	±1599321.199	±1838944.821
40	14683026.77	6960995.12	4354806.94
	±46791.41175	±1710643.963	±1505028.274
41	6732.771297	4483.58	6160.56
	±98.30363249	±212.0746971	±259.3194516
42	19461.39209	10244.34	14808.36
	±209.1069383	±269.4706534	±551.1940567
43	104247.9541	46731.16	43368.64
	±729.867954	±20321.54771	±10886.86633

TABLE VIII. RESULTS (TSP INDEX 44 TO 50)

TSP Index	BSKF	BGSA	BPSO
44	166982.9741	86997.04	110450.06
	±726.4784448	±13437.60017	±23239.6932
45	45944.33039	46345.06	45849.82
	±760.4033317	±635.0789895	±802.5731244
46	8916298.533	5967481.66	5765121.82
	±31536.89852	±706717.6655	±377814.958
47	9302485.934	6391573.08	5612973.5
	±35149.99016	±151284.587	±652337.7034
48	14157633.97	11189186.8	8877423.56
	±49583.4220	±1377344.603	±2608097.778
49	2890.875403	2941.56	2891.9
	±57.40061545	±57.78219484	±57.82353521
50	1409168.891	998335.88	1096777.98
	±13254.61871	±24836.71438	±26460.53055

TABLE IX. AVERAGE RANKINGS

Algorithm	Ranking
BPSO	1.5
BGSA	1.86
BSKF	2.64

TABLE X. WILCOXON TEST RESULT

Comparison	R^+	R^-
BSKF vs BGSA	173	1102
BSKF vs BPSO	20	1255

VI. CONCLUSION

This paper reports the first attempt to use SKF for solving combinatorial optimization problems. The proposed BSKF employed a mapping function to translate the estimated value produced by SKF into probabilistic value. The probabilistic value is then compared to a random value to enable the bit string update at every iteration.

Later, the proposed BSKF could be used in real-world applications which involve discrete and combinatorial optimizations such as DNA sequence design [5], hole drill routing in printed circuit board [6], travelling salesman problem [7], bioinformatics [8], and manufacturing [9].

Note that experimental result and analysis showed that the performance of existing BPSO and BGSA are better than BSKF. Currently, more experiments are being done. In particular, different benchmark functions are considered in order to obtain a more concrete conclusion.

ACKNOWLEDGMENT

This work is financially supported by the UMP Post Graduate Research Scheme (GRS1503120) and by the Fundamental Research Grant Scheme (RDU140132) awarded by the Ministry of Higher Education (MOHE) to Universiti Malaysia Pahang (UMP).

REFERENCES

[1] Z. Ibrahim, N. H. Abdul Aziz, N. A. Ab Aziz, S. Razali, M. I. Shapiai, S. W. Nawawi, and M. S. Mohamad, “A Kalman filter approach for solving unimodal optimization problems,” ICIC Express Letters (accepted), 2015.

[2] J. Kennedy, and R. Eberhart, “A discrete binary version of the particle swarm algorithm.,” IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, Vol. 5, pp. 4104–4108, 1997.

[3] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, “BGSA: binary gravitational search algorithm,” Natural Computing, Vol. 9, Issue 3, pp. 727-745, 2010.

[4] comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[5] Z. Ibrahim, N. K. Khalid, J. A. A. Mukred, S. Buyamin, Z. M. Yusof, M. F. Mohamed Saaid, N. Mokhtar and A. P. Engelbrecht, “A DNA sequence design for DNA computation based on binary vector evaluated particle swarm optimization,” International Journal of Unconventional Computing, Vol. 8, No. 5B, pp. 119-137, 2012.

[6] M. H. Othman, A. F. Zainal Abidin, A. Adam, Z. M. Yusof, Z. Ibrahim, S. M. Mustaza, and L. Y. Yang, “A binary particle swarm optimization approach for routing in PCB holes drilling process,” International Conference on Robotic Automation System, pp. 201-206, 2011.

[7] G. Gutin, A. Yeo, and A. Zverovich, “Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP,” Discrete Applied Mathematics, Vol. 117, No 1–3, pp. 81–86, 2002.

[8] J. X. Yeoh, C. K. Chong, M. S. Mohamad, Y. W. Choon, L. E. Chai, S. Deris, and Z. Ibrahim, “Parameter estimation using improved differential evolution and bacterial foraging algorithms to model tyrosine production in mus musculus (mouse),” Jurnal Teknologi, Vol. 72, No. 1, pp. 49-56, 2015.

[9] I. Ibrahim, Z. Ibrahim, H. Ahmad, M. F. Mat Jusof, Z. M. Yusof, S. W. Nawawi, and M. Mubin, “An assembly sequence planning approach with a rule-based multi-state gravitational search algorithm,” The International Journal of Advanced Manufacturing Technology, Vol. 79, Issue 5, pp. 1363-1376, 2015.