BAB VI PENUTUP

6.3. Kesimpulan

Hasil analisis optimasi dengan menggunakan algoritma MOACO dapat digunakan untuk mencari dan mengoptimasi jalur terbaik dalam mitigasi bencana gunung berapi. Indicator yang dipakai dalam perhitungan algoritma MOACO terbukti sangat baik dalam mengoptimasi jalur pada area bencana. Pada hasil ini, dilakukan pengabaian terhadap kriteria yang nilainya kecil atau nol. Kesimpulan dilakukan setelah mendapatkan optimasi jalur terbaik kemudian ditetapkan jalur alternatifnya. System memberi rekomendasi terhadap jalur utama dan jalur alternatif sebagai pilihan kedua jika jalur utama tidak digunakan.

Hasil analisis algoritma MOACO dapat digunakan sebagai strategy dalam menyelesaikan masalah optimasi mitigasi bencana. Dalam implementasinya, perlu diperhatikan jumlah koloni yang digunakan dalam algoritma dan juga parameter serta jumlah struktur pheromonen dalam algoritma tersebut. Hasil analisis menunjukan pada algitma MOACO sangat efektif dalam melakukan optimasi pada mitigasi bencana pada daerah yang memiliki populasi yang banyak. Algoritma ini dapat digunakan secara efektif dalam membuat peta mitigasi bencana dan jalur yang akan digunakan saat bencana terjadi.

6.3. Rekomendasi

Rekomendasi yang dapat diberikan adalah :

1. Para pemangku kepentingan terutama pemerintah melalui dinas terkait agar dapat menggunakan hasil analisis ini sebagai pantokan dalam mengoptimasi mitigasi bencana di daerah rawan bencana.
2. Informasi peta mitigasi bencana dapat diakses oleh masyarakat umum agar masyarakat mendapat informasi mitigasi yang akurat dan terpercaya.
3. Pentingnya melakukan edukasi kepada masyarakat di erea bencana agar lebih peduli dalam menghadi kejadian bencana dengan mengikuti panduan dan system mitigasi yang telah diatur.

6.3. Disksusi

Dalam penelitian ini, perlu ditindak lanjuti dengan membuat aplikasi yang mudah diakses oleh berbagai pihak seperti operator, pembuat keputusan (pemerintah) dan juga masyarakat umum. Penelitian ini belum membahas tentang aplikasinya dan diharapkan dapat dilanjutkan oleh peneliti yang lain dalam merancang dan membangun aplikasi untuk mitigasi bencana ini. Hal ini disebabkan penelitian ini membahas tentang optimasi yang dapat dilakukan dan dihasilkan oleh algoritma MOACO ini dalam menyeselasikan masalah mitigasi bencana.

REFERENCES

[1] SIBIS, SIBIS New eEurope Indicator Handbook, European Commission publications., 2003.

[2] S. A. A. A. I W. Sudarsana, "Model Matematika Untuk Sistem Evakuasi Tsunami Kota Palu (SET-KP) Berbasis Jalur Terpendek Dan Waktu Evakuasi Minimum," *Online Jurnal of Natural Science,* pp. 39-53, 2013.

[3] M. I. Era Madona, "Aplikasi Metode Nearest Neighbor Pada Penentuan Jalur Evakuasi Terpendek Untuk Daerah Rawan Gempa Dan Tsunami," *Jurnal Elektron Vol 5 No. 2,* pp. 45-51, 2013.

[4] Badan Nasional Penanggulangan Bencana RI, "Data Bencana Indonesia," BNPB, Jakarta, 2015.

[5] S. T. Marco Dorigo, Ant Colony Optimization, London England and Cambridge, Massachusetts: A Bradford Book and The MIT Press, 2004.

[6] X. L. F. S. W. Jin Feng Wang, "A Graph-Based Ant Colony Optimization Approach for Process Planning," *Hindawi Publishing Corporation The Scientific World Journal,* pp. 1-11, 2014.

[7] Y. D. J. G. Liqiang Liu, "Ant Colony Optimization Algorithm for Continuous Domains Based on Position Distribution Model of Ant Colony Foraging," *Hindawi Publishing Corporation The Scientific World Journal,* pp. 1-9, 2014.

[8] I. I. M. López, Multi Objective Ant Colony Optimization, Stanford, California: TECHNISCHE UNIVERSITÄTDARMSTADT and Universidad de Granada, 2004.

[9] E. Verdianto, "Perancangan Sistem Penentuan Rute Terpendek Jalur Evakuasi Tsunami Dengan Algoritma Ant Colony Studi Kasus: Belawan," Ilmu Komputer Universitas Sumatra Utara, Medan, 2013.

[10] A. Leksono, "Algoritma Ant Colony Optimization (Aco) Untuk Menyelesaikan Traveling Salesman Problem (TSP)," Fakultas FMIPA Universitas Diponegoro, Semarang, 2009.

[11] J. R. Batmetan, "Algoritma Ant Colony Optimization (ACO) untuk Pemilihan Jalur Tercepat Evakuasi Bencana Gunung Lokon Sulawesi Utara," *AITI UKSW*, 2016.

[12] R. I. Badan Geologi, "Laporan Aktivitas Gunung Berapi Lokon Sulawesi Utara," Badan Geologi Kementrian ESDM RI, Jakarta, 2015.

[13] B. N. P. B. (BNPB), "Peta Topografi Dan Kawasan Rawan Bencana Gunung Lokon," BNPB, Jakarta, 2011.

[14] P. K. Dharmendra Sutariya, "A Survey Of Ant Colony Based Routing Algorithms For Manet," *European Scientific Journal*, pp. 82-91, 2013.

[15] J. A. A. Fransisca Arvevia I A, "Path Selection In Emergency Evacuation Using Quantum Ant-Colony Algorithm," *Teknik Informatika, Fakultas Teknik, Universitas Telkom*, pp. 1-8, 2014.

[16] T. B. R. P. Chaimongkon Chokpanyasuwan, "Ant Colony Optimization for Load Management Based on Load Shifting in the Textile Industry," *American Journal of Applied Sciences*, pp. 142-154, 2015.

[17] Y. B. Mohammed Taha Benslimane, "Ant Colony Algorithm for the Multi-Depot Vehicle Routing Problem in Large Quantities by a Heterogeneous Fleet of Vehicles," *INFOR*, pp. 31-40, 2013.

[18] C. Daniel Angus, "Multiple Objective Ant Colony Optimisation," *Swarm Intell (2009)*, no. November 2008, pp. 69-85, 2009.

[19] S. T. Manuel López-Ibáñez, "An experimental analysis of design choices of multi-objective ant colony optimization algorithms," *Swarm Intelligence*, vol. 6, no. 3, pp. 207-232, 2012.

[20] M. E. R. Yassine Saji, "Multi-Objective Ant Colony Optimization Algorithm to Solve a Nurse Scheduling Problem," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 8, pp. 311-320, 2013.

[21] M. L.-I. a. T. Stützle, "The Automatic Design of Multi-Objective Ant Colony Optimization Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 6, p. 861–875, 2012.

[22] P. G.-S. J. J. M. P. A. C. A. M. Mora, "Pareto-based multi-colony multi-objective ant colony optimization algorithms: an island model proposal," *Soft Comput*, vol. 17, p. 1175–1207, 2013.

[23] M. L.-I. a. T. Stützle, "The impact of design choices of multiobjective ant colony optimization algorithms on performance: An experimental study on the biobjective TSP," IRIDIA - Technical Report Series, Technical Report No. TR/IRIDIA/2010-003, Bruxelles, Belgium, 2010.

[24] D. T. T. L. e. a. Dickson K. W. Chiu, "Alert based disaster notification and resource allocation," *Inf Syst Front*, vol. 12, p. 29–47, 2010.

[25] e. a. Marco Lu¨tzenberger, "A common approach to intelligent energy and mobility services in a smart city environment," *J Ambient Intell Human Comput*, vol. 6, p. 337–350, 2015.

[26] e. a. Albert Ali Salah, "Multimodal identification and localization of users in a smart environment," *J Multimodal User Interfaces*, vol. 2, p. 75–91, 2008.

[27] X. Zhang, "Development Of A Mixed-Flow Optimization System For Emergency Evacuation In Urban Networks," Faculty of the Graduate School of the University of Maryland, College Park, Maryland, 2012.

[28] H. Lyonnais, "Metode Pencarian Lintasan Terpendek Graf untuk Evakuasi Bencana," *Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung*, pp. 1-5, 2012.

[29] S. S. A. B. R. K. S. C. Jayadeva, "Ants find the shortest path: a mathematical proof," *Swarm Intell*, vol. 7, p. 43–62, 2013.

[30] C. Coffrin, "Decision support for disaster management through hybrid optimization," *Constraints*, vol. 20, p. 490–491, 2015.

[31] M. S. J. e. a. Ali Bozorgi-Amiri, "A modified particle swarm optimization for disaster relief logistics under uncertain environment," *Int J Adv Manuf Technol*, vol. 60, p. 357–371, 2012.

[32] C.-H. Hsu, "Ubiquitous Intelligence and Computing: building smart environment in real and cyber space," *J Ambient Intell Human Comput*, vol. 3, p. 83–85, 2012.

[33] E. E. B. K. Linet Özdamar, "Emergency Logistics Planning in Natural Disasters," *Annals of Operations Research,* vol. 129, p. 217–245, 2004.

[34] V. W. H. N. S. Thushari Silva, "Disaster mitigation and preparedness using linked open data," *J Ambient Intell Human Comput,* vol. 4, p. 591–602, 2013.

[35] P. F. Stefan Propp, "ViSE – A Virtual Smart Environment for Usability Evaluation," *IFIP International Federation for Information Processing,* p. 38–45, 2010.

[36] I. Satyarno, "Earth Quake Disaster Mitigation," Atma Jaya, Yogyakarta, 2016.

[37] W. A. J. B. N. Irwan Iftadi, "Perancangan Peta Evakuasi Menggunakan Algoritma Floyd-Warshall untuk Penentuan Lintasan Terpendek: Studi Kasus," *Performa,* pp. 95-104, 2011.

[38] Y. Z. L. Guo Li, "Comprehensive Optimization of Emergency Evacuation Route and Departure Time under Traffic Control," *The Scientific World Journal,* pp. 1-12, 2014.

[39] I. Fauzy, "Penggunaan Algoritma Dijkstra Dalam Pencarian Rute Tercepat Dan Terpendek, Studi Kasus Pada Jalan Raya Wilayah Blok M Dan Kota," UIN Syarif Hidayatullah, Jakarta, 2011.

[40] M. L.-I. Infante, MultiObjective Ant Colony Optimization, Granda and Stanford: Universidad de Grannada, 2004.

[41] A. A. Z. W. Liu Hongbo, "A Multi-swarm Approach to Multi-objective Flexible Job-shop Scheduling Problems," *Fundamenta Informaticae,* vol. 95, no. 95, pp. 1-25, 2009.

[42] A. I. S. G. A. Chandramohan, "Multi-Objective Optimization Using Ant Colony Optimization for Construction Projects," *National Institute of Technology Calicut, India,* pp. 84-96, 2010.

[43] N. C. W. C. Teerapun Saeheaw, "Application of Ant colony optimization for Multi-objective Production Problems," *World Academy of Science, Engineering and Technology,* vol. 36, pp. 655-660, 2009.

[44] C.-L. Y. Rong-Hwa Huang, "Solving a multi-objective overlapping flow-shop scheduling," *Int J Adv Manuf Technol,* vol. 42, pp. 955-962, 2009.

[45] L. K. V. G. Krasimira Genova, "A Survey of Solving Approaches for Multiple Objective Flexible Job Shop Scheduling Problems," *Cybernetics And Information Technologies,* vol. 15, no. 2, pp. 3-22, 2015.

[46] O. C. F. H. C. Garcı´a-Martı´nez, "A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP," *European Journal of Operational Research,* vol. 180, pp. 116-148, 2007.

[47] C. O. J. F. M. C. Pablo Ortega, "Multiple Ant Colony System For A Vrp With Time Windows And Scheduled Loading," *Ingeniare. Revista chilena de ingeniería,* vol. 17, no. 3, pp. 393-403, 2009.

[48] D. S. Y. J. Sirisha Devi, "Multi Objective Optimization Problem resolution based on Hybrid Ant-Bee Colony for Text Independent Speaker Verification," *I.J. Modern Education and Computer Science,* vol. 1, pp. 55-63, 2015.

[49] R. M. E. L. L. G. A.E. Rizzoli, "Ant colony optimization for real-world vehicle routing problems From theory to applications," *Swarm Intell,* vol. 1, p. 135–151, 2007.

[50] D. D. Vivek S. Borkar, "A novel ACO algorithm for optimization via reinforcement and initial bias," *Swarm Intell,* vol. 3, pp. 3-34, 2009.

[51] M. B. T. S. Z. Y. M. D. Prasanna Balaprakash, "Estimation-based ant colony optimization and local search for the probabilistic traveling salesman problem," *Swarm Intell,* vol. 3, p. 223–242, 2009.

[52] F. N. H. R. C. W. Timo Kötzing, "Theoretical analysis of two ACO approaches for the traveling salesman problem," *Swarm Intell,* vol. 6, pp. 1-21, 2012.

[53] A. M. A. A. A. F. Khalid M. Salama, "Multiple pheromone types and other extensions to the Ant-Miner classification rule discovery algorithm," *Swarm Intell,* vol. 5, p. 149–182, 2011.

[54] T. S. M. B. Paola Pellegrini, "A critical analysis of parameter adaptation in ant colony optimization," *Swarm Intell,* vol. 6, p. 23–48, 2012.

[55] D. I. Candra Dewi, "Map Visualization of Shortest Path Searching of Government Agency Location Using Ant Colony Algorithm," *(IJCSIS) International Journal of Computer Science and Information Security,* vol. 11, no. 11, pp. 19-23, 2013.

[56] Y. D. a. J. G. Liqiang Liu, "Ant Colony Optimization Algorithm for Continuous Domains Based on Position Distribution Model of Ant Colony Foraging," *The Scientific World Journal*, pp. 1-19, 2014.

[57] L. M. A. C. G. C. L. Bacon, "Developing a smart environment for crisis management training," *Ambient Intell Human Comput*, vol. 4, p. 581–590, 2013.

[58] E. A. Nik Bessis, "Preface for a special issue on ‘‘Smart environments and collective computational intelligence for disaster management’’," *J Ambient Intell Human Comput*, vol. 4, p. 533–534, 2013.

[59] T. S. Heiko Rossnagel, "Secure Mobile Notifications of Civilians in Case of a Disaster," *H. Leitold and E. Markatos*, p. 33 – 42, 2006.

[60] G. B. P. F. Stefan Propp, "Task Model-Based Usability Evaluation for Smart Environments," *P. Forbrig and F. Paternò*, p. 29–40, 2008.

[61] J. H. M. P. G. Y. Petri Pulli, "User interaction in smart ambient environment targeted for senior citizen," *Med Biol Eng Comput*, vol. 50, p. 1119–1126, 2012.

[62] T. F. I.D.I.D. Ariyasingha, "A Performance Study for the Multi-objective Ant Colony Optimization Algorithms on the Job Shop Scheduling Problem," *International Journal of Computer Applications*, vol. 132, no. 14, pp. 1-8, 2015.

[63] M. L.-I. a. T. S. Leonardo C. T. Bezerra, "Automatic Generation of Multi-objective ACO Algorithms for the Bi-objective Knapsack," IRIDIA - Technical Report , Series Technical Report No. TR/IRIDIA/2012-013, Bruxelles, Belgium, 2012.

[64] M. L.-I. a. T. Stützle, "An Analysis of Algorithmic Components for Multiobjective Ant Colony Optimization: A Case Study on the Biobjective TSP," IRIDIA - Technical Report Series, Technical Report No. TR/IRIDIA/2009-019, Bruxelles, Belgium, 2009.

[65] M. L.-I. a. T. Stützle, "Automatic Configuration of Multi-Objective ACO Algorithms," IRIDIA - Technical Report Series, Technical Report No. TR/IRIDIA/2010-011, Bruxelles, Belgium, 2010.

[66] E. F. G. M. C. G. L. S. B. Leonardo C.T. Bezerra, "Analyzing the impact of MOACO components: An algorithmic study on the multi-objective shortest path problem," *Expert Systems with Applications*, vol. 40, p. 345–355, 2013.

LAMPIRAN
Source Code
File : moaco.h

```
/************************************************************************
 MOACO
 ---------------------------------------------------------
 Copyright (c) 2005, Manuel Lopez-Ibanez
 TeX: \copyright 2005, Manuel L{\'o}pez-Ib{\'a}{\~n}ez

 This program is free software (software libre); you can redistribute
 it and/or modify it under the terms of version 2 of the GNU General
 Public License as published by the Free Software Foundation.

 This program is distributed in the hope that it will be useful, but
 WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 General Public License for more details.

  You should have received a copy of the GNU General Public License
  along with this program; if not, you can obtain a copy of the GNU
  General Public License at:
             http://www.gnu.org/copyleft/gpl.html
  or by writing to:
        Free Software Foundation, Inc., 59 Temple Place,
            Suite 330, Boston, MA 02111-1307 USA
************************************************************************/
#ifndef _MOACO_H_
#define _MOACO_H_

#include "solution_moaco.h"
#include "solution_wls.h"
#include "libmisc.h"
#include "pareto.h"

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <assert.h>
#include <string.h>
#include <limits.h>

extern const int NUMB_OBJ;
/* #ifndef NUM_OBJ */
/* #define NUM_OBJ 2 */
/* #elif NUM_OBJ != 2 */
```

```
/* #error NUM_OBJ should be 2 */
/* #endif */

#define METAH_NAME "MOACO"

#define STOP_CONDITION (Iteration != Number_Iterations && Timer_elapsed_virtual () < Time_Limit)

extern problem_t * problem;

typedef t_solution * dl_solution_node_t;
typedef struct {
   dl_solution_node_t * node;
   int (*cmp)(const void*, const void*);
   int size;
   int max_size;
} dl_solution_t;

typedef struct {
   int num_ants;
   int *num_ants_per_weight;
   t_solution **ants;
   pheromone_t ph1;
   pheromone_t ph2;
   pheromone_t tau_total;
   pheromone_t dtau1;
   pheromone_t dtau2;
   int *weights;
   t_Pareto *pareto_set;
   dl_solution_t **iteration_best1;
   dl_solution_t **iteration_best2;
   dl_solution_t **best_so_far1;
   dl_solution_t **best_so_far2;
   dl_solution_t **update_best1;
   dl_solution_t **update_best2;
} ant_colony_t;

double * FloatWeights;
ant_colony_t * Colonies;

enum {
   UPDATE_BY_ORIGIN  = 0,
   UPDATE_BY_REGION  = 1
};
static const param_select_type
COLONY_UPDATE_ALTERNATIVES[]  = {
   { "origin", UPDATE_BY_ORIGIN  },
   { "region", UPDATE_BY_REGION  },
   { NULL,  -1 }
};
```

```
// Selection Method Options
enum {
   SELECT_BY_DOMINANCE  = 0,
   SELECT_BY_OBJECTIVE  = 1,
   SELECT_BY_WEIGHT  = 2,
};
static const param_select_type
SELECT_BY_ALTERNATIVES[]  = {
   { "dominance", SELECT_BY_DOMINANCE  },
   { "objective", SELECT_BY_OBJECTIVE  },
   { "weight",   SELECT_BY_WEIGHT  },
   { NULL,  -1 }
};

enum Update_best_ants_t Update_best_ants;
/* enum Update_best_ants_t { */
/*    UPDATE_ANTS_ITERATION_BEST  = 0, */
/*    UPDATE_ANTS_BEST_SO_FAR,  */
/*    UPDATE_ANTS_MIXED_SCHEDULE  */
/* } Update_best_ants; */

static const param_select_type
PARAM_UPDATE_BEST_ALTERNATIVES[]  = {
   { "iteration-best", UPDATE_ANTS_ITERATION_BEST  },
   { "best-so-far", UPDATE_ANTS_BEST_SO_FAR  },
   { "mixed",  UPDATE_ANTS_MIXED_SCHEDULE  },
   { "ib", UPDATE_ANTS_ITERATION_BEST  },
   { "bf", UPDATE_ANTS_BEST_SO_FAR  },
   { NULL,  -1 }
};

enum ACO_algorithm_t  {
   MMAS = 0,
   ACS
} ACO_algorithm;

static const param_select_type
PARAM_ACO_ALGORITHM_ALTERNATIVES[]  = {
   { "mmas",  MMAS },
   { "acs", ACS },
   { NULL,  -1 }
};

// Update Method Options
/* PARAM_DEFINE_ALTERNATIVE  (UPDATE_BY,  */
/*    PARAM_ALTERNATIVE  ("origin", ORIGIN),  */
/*    PARAM_ALTERNATIVE  ("region", REGION)  */
/* ); */

bool MOAQ_flag;
bool PACO_flag;
```

```
bool BicriterionAnt_flag;
bool MONACO_flag;
bool mACO1_flag;
bool mACO2_flag;
bool mACO3_flag;
bool mACO4_flag;
bool MACS_flag;
bool COMPETants_flag;
bool flag_dtau_objective_function;
bool flag_update_only_once;


/* Parameters */
int Num_ants, /* per colony */
    Num_colonies,
    Num_Weights,
    Number_Trials,
    Number_Iterations,
    Ants_candlist_size, /* number of elements in the candidate list for construction */
    LS_candlist_size, /* number of elements in the candidate list for local search */
    Num_update, /* number of solutions used for update. */
    AllWeights_flag,
    SelectMethod, UpdateMethod,
    Quiet;

bool MultiplePheromone_flag,
    MultipleHeuristic_flag,
    LocalSearch_flag;

int Max_Weight;

bool ParetoLocalSearch_flag;
bool ePLS_flag;
double Allowable_Tolerance;
bool WROTS_flag;
int  TabooSearch_Length;
bool Weighted_local_search_flag;
int LocalSearch_type;

double Rho, Time_Limit,  Prob_best, q_0;
double Alpha, Beta;
extern double ph1_min, ph1_max,  ph1_0, ph2_min, ph2_max,  ph2_0;


/* Candidate List */
extern int **Ants_candlist;
int Ants_candlist_size;
int LS_candlist_size;

#define PARETO_SIZE  1000
t_Pareto * BestSoFarPareto;
```

```
t_Pareto * IterationPareto;
//t_Pareto * RestartPareto;

unsigned long Seed;

FILE *Report;
FILE *Trace;

int Iteration, Trial;

double time_localsearch;

static inline void
trace_header (int current_trial)
{
  DEBUG2_PRINT ("start_trial(%d) :\n", current_trial);

  if (Trace) {
    fprintf (Trace,
        "# start_trial(%d) :\n"
        "# Iterat  Add  Rmv  Size    Time\n",
        current_trial);
  }
}
static inline void
trace_print (int iteration_found_best, int added, int removed, int size,
       double time)
{
  DEBUG2_FUNPRINT ("iteration_found_best = %6d, added = %4d, removed = %4d,"
          " size_bf = %4d, time = %8.8g\n",
          iteration_found_best, added, removed, size, time);
  if (Trace && Iteration % 100) {
    fprintf (Trace, " %6d %4d %4d %4d %8.8g\n",
        iteration_found_best, added, removed, size, time);
  }
}

void dl_solution_clear (dl_solution_t * list);

/* moaco_io.c */
void parameter_defaults (void);
void read_parameters(int argc, char **argv);
void write_parameters(FILE *stream, const char *str, int argc, char *argv[]);
void setup_weights(int **candlist, int candlist_size);
void report_print_header (int argc, char *argv[]);
void start_trial( int actual_trial );
void end_trial( int actual_trial, int actual_iteration );
void end_program (void);
#endif
```

File : moaco_parameters.h

```
DEFINE_PARAMETER(PARAM_HELP,
"-h", "--help",      "Prints this information and exits")
DEFINE_PARAMETER(PARAM_VERSION,
"-v", "--version",   "Prints version and exits")
DEFINE_PARAMETER(PARAM_INPUT,
"-i", "--input",     "FILE     Data file")
DEFINE_PARAMETER(PARAM_SEED,
"-s", "--seed",      "INT:>0    Seed for random number generator")
DEFINE_PARAMETER(PARAM_OUTPUT,
"-o", "--output",    "FILE     Report file")
DEFINE_PARAMETER(PARAM_TRIALS,
"-r", "--trials",    "INT:>0    Number of trials to be run on one instance")
DEFINE_PARAMETER(PARAM_TIME,
"-t", "--time",      "REAL:>0   Time limit of each trial (seconds)")
DEFINE_PARAMETER(PARAM_ITERATIONS,
"-n","--iterations","INT:>0    Number of iterations of each trial")
DEFINE_PARAMETER(   PARAM_EVALUATIONS,
"-e","--evaluations","INT:>0    Number of evaluations of each trial")
DEFINE_PARAMETER(   PARAM_QUIET,
            "-q", "--quiet",     "Minimum output on report")
DEFINE_PARAMETER(   PARAM_TRACEFILE,
            "-T", "--trace",   "File to report extra information")
DEFINE_PARAMETER(   PARAM_NUMANTS,
            "-m", "--ants", "INT:>0   Number of ants (per colony)")
DEFINE_PARAMETER(   PARAM_TOTALANTS,
            NULL,   "--total-ants", "INT:>0    Number of ants (must be divisible by the number of
colonies)")

DEFINE_PARAMETER(   PARAM_RHO,
            "-p", "--rho", "REAL:[0,1]    Evaporation factor tau(t+1) = (1 - rho) * tau(t)")
DEFINE_PARAMETER(   PARAM_ALPHA,
            "-a", "--alpha", "REAL:[0,1]    Pheromone Alpha factor")
DEFINE_PARAMETER(   PARAM_BETA,
            "-b", "--beta", "REAL:[0,1]    Heuristic Beta factor")
DEFINE_PARAMETER(   PARAM_PROB_BEST,
            "-P", "--prob-best", "REAL:[0,1]    Probability of constructing the best solution")
DEFINE_PARAMETER(   PARAM_Q0,
            "-q0", "--q0", "REAL:[0,1]    Probability of best choice in tour construction (q_0)")
DEFINE_PARAMETER(   PARAM_DTAU,
            NULL,   "--dtau", "Enable objective function value update")
DEFINE_PARAMETER(   PARAM_UPDATE_BEST,
            "-B", "--update-best", "[ iteration-best | best-so-far | mixed  ] Which ants are used for update")
DEFINE_PARAMETER(   PARAM_ANTS_CANDIDATE_LIST,
            "-g", "--candlist", "INT:>=0   Size of the candidate list for construction")

DEFINE_PARAMETER(   PARAM_COLONIES  ,
            "-c", "--colonies", "INT:>0    Number of colonies")
DEFINE_PARAMETER(   PARAM_COLONY_WEIGHTS,
            NULL,   "--colony-weights", "[ same | disjoint | overlapping ] weight intervals for each
colony")
```

```
DEFINE_PARAMETER( PARAM_COLONY_UPDATE,
        NULL, "--colony-update", "[ origin | region ] update method for multiple colonies")
DEFINE_PARAMETER( PARAM_NUM_UPDATE,
                "-u", "--num-update", "INT:>=0 Number of solutions used in update")
DEFINE_PARAMETER( PARAM_SELECT,
        "-S", "--selection", "[ dominance | objective | weight ] Comparison criteria for the selection method")
DEFINE_PARAMETER( PARAM_NUM_WEIGHTS,
        "-w", "--weights", "Number of weights (0 < --weights < --ants). Values within [0.0, 1.0] use Num_ants / value, negative values use Num_ants, positive values larger than 1 use that number of weights.")
DEFINE_PARAMETER( PARAM_DIRECTION,
        "-d", "--directions", "[ one | all ] use one or all weights per iteration")

DEFINE_PARAMETER( PARAM_SINGLE_PHEROMONE,
        NULL, "--ph=single", "Use single pheromone matrix")
DEFINE_PARAMETER( PARAM_MULTIPLE_PHEROMONE,
        NULL, "--ph=multiple", "Use multiple pheromone matrices")
DEFINE_PARAMETER( PARAM_MULTIPLE_HEURISTIC,
        NULL, "--heu=multiple", "Use multiple heuristic matrices")
DEFINE_PARAMETER( PARAM_SINGLE_HEURISTIC,
        NULL, "--heu=single", "Use single heuristic matrix")

DEFINE_PARAMETER( PARAM_AGGREGATION,
        NULL, "--aggregation", "[ sum | product | random ] Method for aggregating multiple matrices")

DEFINE_PARAMETER( PARAM_PHEROMONE_AGGREGATION,
        NULL, "--ph-aggregation", "[ sum | product | random ] Method for aggregating multiple pheromone matrices")

DEFINE_PARAMETER( PARAM_HEURISTIC_AGGREGATION,
        NULL, "--heu-aggregation", "[ sum | product | random ] Method for aggregating multiple heuristic matrices")

DEFINE_PARAMETER( PARAM_ACO_ALGORITHM,
        "-A", "--aco", " [ mmas | acs ] Underlying ACO algorithm")

DEFINE_PARAMETER( PARAM_MOAQ,
        NULL, "--MOAQ", "MOAQ settings")
DEFINE_PARAMETER( PARAM_PACO,
        NULL, "--PACO", "Pareto ACO settings")
DEFINE_PARAMETER( PARAM_BICRITERIONANT,
        NULL, "--BicriterionAnt", "BicriterionAnt ACO settings")
DEFINE_PARAMETER( PARAM_MONACO,
        NULL, "--MONACO", "Multi-Objective Network ACO settings")
DEFINE_PARAMETER( PARAM_mACO1,
        NULL, "--mACO1", "m-ACO variant 1 (m+1,m)")
DEFINE_PARAMETER( PARAM_mACO2,
        NULL, "--mACO2", "m-ACO variant 2 (m+1,m)")
DEFINE_PARAMETER( PARAM_mACO3,
```

```
            NULL, "--mACO3", "m-ACO variant 3 (1,1)")
DEFINE_PARAMETER( PARAM_mACO4,
            NULL, "--mACO4", "m-ACO variant 4 (1,m)")
DEFINE_PARAMETER( PARAM_MACS,
            NULL, "--MACS", "MACS settings")
DEFINE_PARAMETER( PARAM_COMPETants,
            NULL, "--COMPETants", "COMPETants settings")
/* FIXME: Move all these parameter to problem-specific code. */
/*
DEFINE_PARAMETER( PARAM_PLS,
            "-pls", "--paretolocalsearch", "enables Pareto Local Search")
DEFINE_PARAMETER( PARAM_WROTS,
            "-wrots", "--wrots", "INT:>0    enables W-RoTS. Length as multiple of instance size")
DEFINE_PARAMETER( PARAM_ePLS,
            "-epls","--epsilon-pls","INT:>0 enables ePLS and sets the limit in the number of solutions")
*/
DEFINE_PARAMETER( PARAM_LS_CANDIDATE_LIST,
            "-k", "--ls-candlist", "INT:>=0  Size of the candidate list for local search")
#include "solution_wls_parameters.h"
```

File: solution.h

```
#ifndef _SOLUTION_H_
#define _SOLUTION_H_

#define FILENAME_LEN  256

extern const int NUM_OBJ;
#include "libmisc.h"

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <error.h>
#include <assert.h>

#include "t_number_def.h"
#include "btsp.h"

typedef t_btsp_instance problem_t;

typedef struct {
   int *permutation;
   t_number o[2];
   /* FIXME: these should belong to ant_type not to solution_type. */
   int colony; // Original colony (for MOACO)
   int weight_idx; // Weight index used to generate this solution (for MOACO)
   double weight; // Real weight used to generate this solution (for MOACO)
} t_solution;

typedef t_solution *t_Solution;
```

```
problem_t * SolInitProblem(const  char * inp_filename);
static inline int problem_get_size (const problem_t * problem) {
   return problem->size;  }

t_solution * SolCreate(void);
void SolEvaluate (t_solution *);
void SolCheck (const t_solution *);
void SolCopy(t_solution *dest, const t_solution *scr);
int Solcmp(const void * s1, const void * s2); // for qsort() and alike
int Solcmp_obj1(const void * p1, const void * p2);
int Solcmp_obj2(const void * p1, const void * p2);
int SolCompare_obj1(const t_solution * p1, const t_solution * p2);
int SolCompare_obj2(const t_solution * p1, const t_solution * p2);
int SolCompare(const t_solution * s1, const t_solution * s2);

/* FIXME:  this should return bool
   1 -> if a dominates b
   0 -> otherwise.
   Another function SolDominance(a,b) should return:
   -1 -> if a dominates b
   0  -> if a == b
   1 -> if  a not dominates b
*/
static inline int
SolDominates(const t_solution *a, const t_solution *b)
/*********************************************
 return: 0  -> if a == b
        1  -> if a dominates b
        -1 -> if a NOT dominates b

*********************************************/
{
   //If any objective is worse, A can't dominate B
   //    for(i=0;  i < NUM_OBJ;  i++)
   if (a->o[0] > b->o[0]) return -1;
   if (a->o[1] > b->o[1]) return -1;

   // If any objective is better, then A dominates B
   //    for(i=0;  i < NUM_OBJ;  i++)
   if (a->o[0] < b->o[0]) return 1;
   if (a->o[1] < b->o[1]) return 1;
   // Otherwise A == B
   return 0 ;
}

void SolFree(t_solution *s);
t_number SolGenerate_greedy_with_weight (t_solution *solution, t_number weight, t_number
max_weight);
void SolGenerateRandom(t_solution *s);

static inline int SolInfeasibility (const t_solution *s __unused) { return 0; }
```

```
static inline bool SolIsInfeasible (const t_solution *s __unused) { return 0; }

static inline
int * SolGetVector(const  t_solution * s)
{
   return s->permutation;
}

static inline t_number
SolGetObjective(const  t_solution *s, int obj)
{
#if DEBUG  > 0
   if (obj <= 0 || obj > NUM_OBJ)  {
      fprintf (stderr, "%s(): objective %d does not exist!\n", __FUNCTION__,
            obj);
      exit (EXIT_FAILURE);
   }
#endif
   return s->o[obj-1];
}

static inline void SolSetObjective(t_solution *s, int obj, t_number valor)
{
   assert (obj == 1 || obj == 2);
   s->o[obj - 1] = valor;
}

static inline void SetSolutionColony(t_solution * s, int c)
{
   s->colony = c;
}

static inline int GetSolutionColony(const t_solution * s)
{
   return s->colony;
}

void SolPrint(FILE   *stream, const t_solution *s);
void SolPrintOneLine(FILE  *stream, const t_solution *s);
void SolPrintParam(FILE   *stream, const char *prefix);
void SolPrintVersion(FILE   *stream, const char *prefix);

void SolProblemSetWeights  (const t_number *weights,
                     int num_weights,
                     int **candlist,
                     int candlist_size);
int **SolParetoCandList  (int candlist_size);
int **SolCombinedCandList  (int candlist_size);

static inline int Sol_get_weight_idx (const t_solution *s)
{
```

```
   return s->weight_idx;
}
static inline void Sol_set_weight_idx (t_solution *s, int w)
{
   s->weight_idx = w;
}

static inline double Sol_get_weight (const t_solution *s)
{
   return s->weight;
}
static inline void Sol_set_weight (t_solution *s, double w)
{
   s->weight = w;
}

/* Not actually used. */
int ***
SolWeightedCandidateList (int candlist_size, int num_weights);

/* Private */
void
calc_weighted_matrix (t_number **, t_number **m1, t_number **m2, int n,
                      t_number weight1, t_number max_weight);

#endif
```