SOP Challenge - Presentation

Sebastian ALLMEIER
Félix CARON
Ionut-Vlad MODORANU

January 7, 2020

Sebastian ALLMEIER
Félix CARON
Ionut-Vlad MODORANU

Exact Method
Greedy Method
Beam Search
DPSO
Results

Structure

1. Exact Method
2. Greedy Method
3. Beam Search
4. DPSO
5. Results

Exact Method

Data

V	:	Set of vertices.
w_{ij}	:	Weight of the arc going from vertex i to j.
C	:	Set of precedence constraints.
		If $(i, j) \in C$, i must be visited before j.
s	:	Start vertex.
t	:	Destination vertex.

Decision variables

$$x_{ij} = \begin{cases} 1 & \text{if vertex } i \text{ is followed by } j \\ 0 & \text{otherwise} \end{cases}$$

y_{ij} : Number of packages traveling from vertex i to j.

Objective function

$$\min \sum_{i \in V} \sum_{j \in V} w_{ij} x_{ij}$$

Exact Method

Constraints

$$\sum_{j \in V} x_{ij} = 1, \qquad \forall i \in V$$

$$\sum_{i \in V} x_{ij} = 1, \qquad \forall j \in V$$

$$x_{ts} = 1$$

$$x_{ii} = 0, \qquad \forall i \in V$$

$$\sum_{j \in V} y_{sj} = |V| - 1$$

$$y_{ij} \leq (|V| - 1)x_{ij}, \qquad \forall i, j \in V$$

$$\sum_{j \in V} y_{ij} - \sum_{j \in V} y_{ji} = -1, \qquad \forall i \in V \setminus \{s\}$$

$$\sum_{k \in V} y_{kj} - \sum_{k \in V} y_{ki} \geq 0, \qquad \forall (i,j) \in C \mid i \neq s$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in V$$

$$y_{ij} \in \mathbb{N}, \qquad \forall i, j \in V$$

Greedy Method

- initialize path with the initial node
- compute all children of the current path (respecting the precedence constraints)
- select the best one
- repeat until end node is reached

Beam Search

General Idea

- specify the beam width $K \in \mathbb{N}$
- initialize one path containing the initial node
- compute all children from the current paths (respecting precedence constraints)
- select the K best ones
- repeat until end node is reached (all nodes have been visited)

Sebastian
ALLMEIER
Félix CARON
Ionut-Vlad
MODORANU

Exact Method

Greedy
Method

Beam Search

DPSO

Results

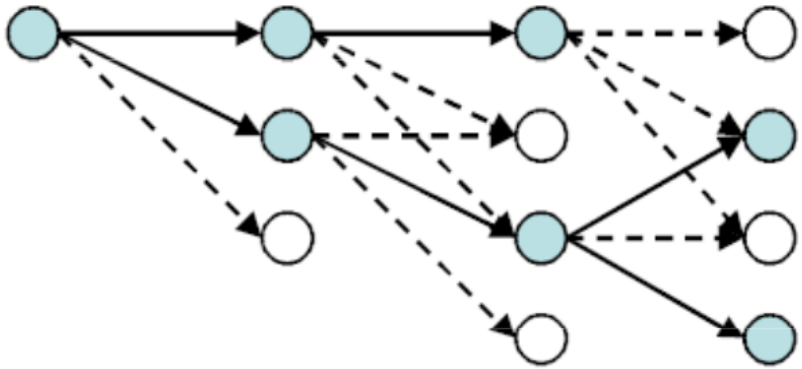

Figure: Example of a beam search tree [1]

Particle Swarm Optimization

- Used for multimodal functions
- Population based
- Iterative algorithm

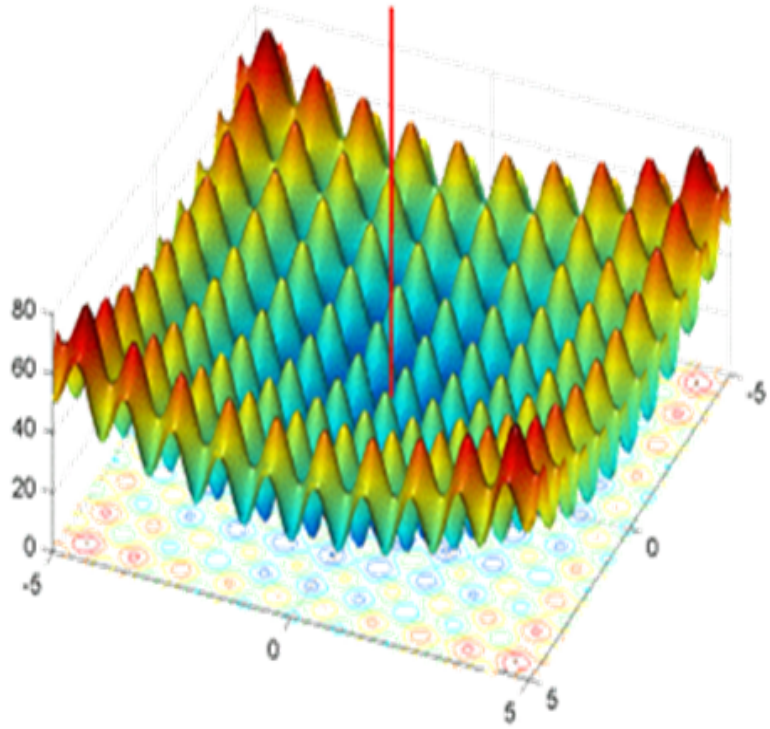

Figure: Example of multimodal function - Rastrigin

Particle Swarm Optimization

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{inertia}} + \underbrace{c_1 U_1^t (\mathbf{pb}_i^t - \mathbf{p}_i^t)}_{\text{personal influence}} + \underbrace{c_2 U_2^t (\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{social influence}}$$

$$\mathbf{p}_i^{t+1} = \mathbf{p}_i^t + \mathbf{v}_i^{t+1}$$

Figure: PSO update formula

- Particles are vectors of coordinates
- Velocities are coordinate-wise
- Feasibility of particles is guaranteed

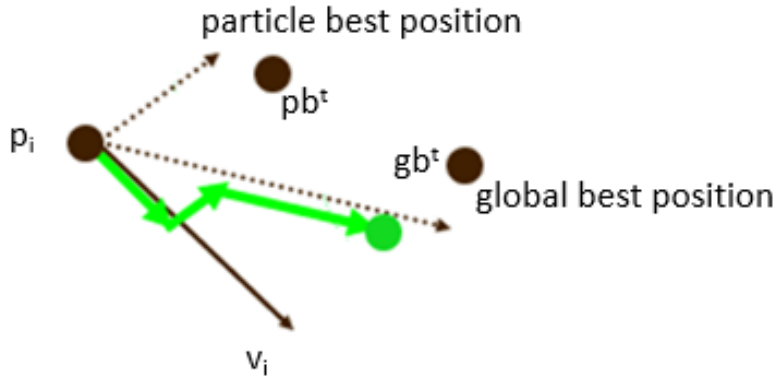

Figure: PSO update

Discrete Particle Swarm Optimization

- Particles are permutations
- Velocities are pairs (n, d):
  - $n =$ node
  - $d =$ left/right displacement

$$\mathbf{v}_i^{t+1} = \underbrace{\mathbf{v}_i^t}_{\text{inertia}} + \underbrace{c_1 U_1^t (\mathbf{pb}_i^t - \mathbf{p}_i^t)}_{\text{personal influence}} + \underbrace{c_2 U_2^t (\mathbf{gb}^t - \mathbf{p}_i^t)}_{\text{social influence}}$$

$$\mathbf{p}_i^{t+1} = \mathbf{p}_i^t + \mathbf{v}_i^{t+1}$$

- $pA - pB$: velocities to pB to obtain pA
- $const * velocity$: delete or modify elements in $velocity$
- $perm + velocity$: modify $perm$ with moves in $velocity$

Feasibility of particles is NOT guaranteed $\Rightarrow$ modify permutation to respect precedence constraints (very slow)

Solution Costs

Instance	LB	UB	DPSO	Greedy	BS10	BSV
ESC07	2125	2125	2125	2700	2125	2125
ESC11	2075	2075	2195	3175	2480	2480
ESC12	1675	1675	1675	2034	2429	1997
ESC25	1681	1681	6518	3360	2010	1821
ESC47	1288	1288	11617	3843	2186	2664
ESC63	62	62	185	76	72	67
ESC78	18230	18230	27900	22600	20425	20340
kro124p.1	38762	39420	-	52575	49074	49680
kro124p.2	39841	41336	-	57723	54185	52568
kro124p.3	43904	49499	-	77266	64330	60999
kro124p.4	73021	76103	-	98427	94773	89152
ry48p.1	15805	15805	39933	22493	17739	18888
ry48p.2	16074	16666	40867	20911	18829	19207
ry48p.3	19490	19894	40134	27342	24703	24309
ry48p.4	31446	31446	39376	41176	38639	38488
R.500.1000.1	1316	1316	-	6205	5397	4306
R.500.1000.15	43134	49504	-	111129	63321	51582
R.500.1000.30	98987	98987	-	155387	113208	-
R.500.1000.60	178212	178212	-	205604	180442	-
R.600.1000.1	1337	1337	-	4931	5523	-
R.600.1000.15	47042	55213	-	120975	71601	-
R.600.1000.30	126789	126789	-	189988	136791	-
R.600.1000.60	214608	214608	-	256253	222597	-
R.700.1000.1	1231	1231	-	4886	5369	-
R.700.1000.15	54351	65305	-	151331	82151	-
R.700.1000.30	134474	134474	-	208460	149117	-
R.700.1000.60	245589	245589	-	277504	250512	-

Computation Times

Instance	DPSO	Greedy (in s)	BS10 (in s)	BSV (in s)
ESC07	3m 38s	0.000	0.003	0.002
ESC11	7m 2s	0.001	0.009	0.009
ESC12	8m 15s	0.001	0.009	0.014
ESC25	30m 28s	0.004	0.079	0.283
ESC47	3h 2m 31s	0.015	0.415	2.367
ESC63	50m 3s	0.026	1.074	8.499
ESC78	22m 3s	0.036	1.225	9.83
kro124p.1	50m 17s	0.062	4.307	51.46
kro124p.2	-	0.080	3.833	56.224
kro124p.3	-	0.077	2.855	34.347
kro124p.4	-	0.076	1.445	17.222
ry48p.1	19m 42s	0.016	0.61	2.735
ry48p.2	6m 5s	0.020	0.651	2.524
ry48p.3	1h 26m 39s	0.018	0.442	2.027
ry48p.4	1h 6m 18s	0.015	0.269	1.071
R.500.1000.1	-	2.617	591.266	33420 (9h17m)
R.500.1000.15	-	25.288	222.86	11820 (3h17m)
R.500.1000.30	-	25.989	224.676	-
R.500.1000.60	-	26.161	199.452	-
R.600.1000.1	-	4.022	1005.034	-
R.600.1000.15	-	44.231	376.847	-
R.600.1000.30	-	43.502	393.982	-
R.600.1000.60	-	43.993	361.164	-
R.700.1000.1	-	5.865	1723.453	-
R.700.1000.15	-	68.580	598.029	-
R.700.1000.30	-	69.449	616.624	-
R.700.1000.60	-	69.626	679.959	-

S. Lazebni, "Local search algorithms." https://www.cs.unc.edu/~lazebnik/fall10/lec06_local_search.pdf.
Accessed on 2020-01-03.