Standard Swarm Optimization to Solve Machine Scheduling Problem with Fuzzy Starting Time

Y. M. Assem[1], A. A. El-sawy[2] and Y. A. Nada[3]

[1,3] Computer Engineering Dept., Faculty of Computers and Information Technology, Taif University Taif, KSA.

[2] Computer Science Dept., Faculty of Computers and Information, Benha University Egypt.

Abstract

One of machine scheduling problems called machine time scheduling. The starting time in this problem will be assumed fuzzy with trapezoid membership function. A standard particle swarm optimization algorithm with constriction factor has been developed to find the best shape of membership function to find best starting time for each machine in each cycle by mean max defuzzification method.

Keywords: *Machine Time Scheduling, Particle Swarm optimization, Fuzzy, Time Window, Trapezoid membership function, mean max defuzzification method.*

1. Introduction

A great deal of research has been focused on solving scheduling problems. One of the most important scheduling problems is the Machine Time Scheduling Problem (MTSP). This problem was investigated in [14] as a parameterized version of the MTSP, which was defined in [8], with penalized earliness in starting and lateness in the completion of the operation. The authors in [14] applied the optimal choice concept which is given in [11] and some theoretical results from [12] to obtain the optimal values of the given parameters.

In [4] the authors investigated two cycles MTSP and introduced an algorithm to find the optimal choice of parameters, which represent the earliest possible starting time for the second cycle. In [3] an algorithm was developed (MTSP Algorithm (MTSPA)) for multi-cycles MTSP which found the starting time for each machine in each cycle by using the max-separable technique. The processing times in the previous researches were deterministic. A generalization was introduced in [2] to overstep the cases at which an empty feasible set of solutions is described by the system.

In [1] introduced an algorithm by using the PSO and GA to solve MTSP, and compared between PSO, GA and max-separable technique (using numerical example). The Authors found that PSO algorithm reach to the best solution than GA and max-separable technique. In [9] discusses how to solve the MTSP when the processing time for each machine is stochastic. To solve this problem, the Monte Carlo simulation is suggested to handle the given stochastic processing times. In this paper we will introduce an algorithm to find best starting time for each machine in each cycle when starting time is fuzzy. We assumed that the starting time follow trapezoid memberships function. We will develop an algorithm by using particle swarm optimization algorithm (with constriction factor) to adapt the shape of the membership function for each starting time for each machine then mean max defuzzification method well be used to find the crisp value.

2. Problem Formulation

In machine time scheduling problem there are n machines, each machine carries out one operation j with processing time p_j for $j \in N = \{1,...,n\}$ and the machines work in k cycles. Let $\tilde{x}_{jr}$ represent starting time of the j^{th} machine in cycle r for all $j \in N$, $r \in K = \{1,...,k\}$ (k number of cycles) and is random variable with certain distribution. Machine j can start its work in cycle r only after the machines in a given set $N^{(j)}, N^{(j)} \subset N$ ($N^{(j)}$ is the set of precedence machines) had finished their work in the $(r\text{-}1)^{th}$ cycle, so we can define the starting time in the $(r\text{-}1)^{th}$ cycle as follows:

$$\tilde{x}_{ir+1} \geq \max_{j \in N^{(i)}} (x_{jr} + p_{jr}) \quad \forall i \in N, \forall r \in K \qquad (1)$$

Assuming that the starting time x_{jr} is constrained by a time interval $[l_{jr}, L_{jr}]$ for each $j \in N$, $r \in K$ and, then the set of feasible starting times $\tilde{x}_{jr}$ is described by the following system for each $r \in K$:

$$\begin{aligned} &\max_{j \in N^{(i)}} (\tilde{x}_{jr} + p_{jr}) \leq \tilde{x}_{ir+1} \quad \forall i \in N, \\ &l_{jr} \leq \tilde{x}_{jr} \leq L_{jr} \qquad \forall j \in N \end{aligned} \qquad (2)$$

Assume also that for some echological reasons there are a given recommended time interval $[a_{jr}, b_{jr}]$, $\forall i \in N$, $\forall r \in K$ so:

$$[\tilde{x}_{jr}, \tilde{x}_{jr} + p_j] \subset [a_{jr}, b_{jr}], \qquad (3)$$

The violation of the Eq. (3) will be penalized by the following penalty function

$$f(\tilde{x}) = \max_{j \in N} f_{jr}(\tilde{x}_{jr}) \rightarrow \min \qquad r \in K \qquad (4)$$

Where the penalty function in a certain cycle r is given by:

$f_{jr}(\tilde{x}_{jr}) = \max\{ f_{jr}^{(1)}(\tilde{x}_{jr}), f_{jr}^{(2)}(\tilde{x}_{jr} + p_{jr}), 0\} \quad \forall j \in N$ Wher e $f_{jr}^{(1)}$: R→ R is decreased continuous function such that $f_{jr}^{(1)}(a_{jr})=0$,

And $f_{jr}^{(2)}$: R→ R is increasing continuous function such that $f_{jr}^{(2)}(b_{jr})=0$

To minimize the maximum penalty in each cycle r, we should solve the following problem:

$f(\tilde{x}) \rightarrow \min$

subject to :

$$\max_{j \in N^{(i)}} (\tilde{x}_{jr} + p_{jr}) \le \tilde{x}_{ir+1} \quad \forall i \in N \qquad (5)$$

$$l_{jr} \le \tilde{x}_{jr} \le L_{jr} \qquad \forall j \in N$$

3. Particle Swarm Optimization

The PSO method is a member of wide category of Swarm Intelligence methods for solving the optimization problems. It is a population based search algorithm where each individual is referred to as particle and represents a candidate solution. Each particle in PSO flies through the search space with an adaptable velocity that is dynamically modified according to its own flying experience and also the flying experience of the other particles. Further, each particle has a memory and hence it is capable of remembering the best position in the search space ever visited by it. The position corresponding to the best fitness is known as *pbest* and the overall best out of all the particles in the population is called *gbes*t [10].

The modified velocity and position of each particle can be calculated using the current velocity and the distance from the $pbest_j$ to *gbest* as shown in the following formulas:

$$v_{j,g}^{(t+1)} = w * v_{j,g}^{(t)} + c_1 * r_1 * (pbest_{j,g} - x_{j,g}^{(t)}) + c_2 * r_2 * (gbest_{j,g} - x_{j,g}^{(t)}) \qquad (6)$$

$$x_{j,g}^{(t+1)} = x_{j,g}^{(t)} + v_{j,g}^{(t+1)} \qquad (7)$$

With $j=1, 2, \ldots, n$ and $g=1, 2, \ldots, m$

n =number of particles in a group;

m = number of members in a particle;

t = number of iterations (generations);

$v_{j,g}^{(t)}$ =velocity of particle j at iteration t,

w = inertia weight factor;

c_1, c_2 = cognitive and social acceleration factors, respectively;

r_1, r_2 = random numbers uniformly distributed in the range (0, 1);

$x_{j,g}^{(t)}$ = current position of j at iteration t;

$pbest_j$ = *pbest* of particle j;

gbest = *gbest* of the group.

The index of best particle among all of the particles in the group is represented by the *gbest*. In PSO, each particle moves in the search space with a velocity according to its own previous best solution and its group's previous best solution. The velocity update in a PSO consists of three parts; namely momentum, cognitive and social parts. The balance among these parts determines the performance of a PSO algorithm. The parameters c_1 & c_2 determine the relative pull of *pbest* and *gbest* and the parameters r_1 & r_2 help in stochastically varying these pulls [10]. [6] Showed that combining them by setting the inertia weight, ¡ , to the constriction factor, v, improved performance across a wide range of problems as follows:

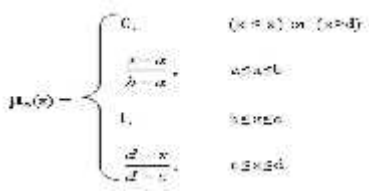
$$v_{j,g}^{(t+1)} = ¡\{w * v_{j,g}^{(t)} + c_1 * r_1 * (pbest_{j,g} - x_{j,g}^{(t)}) + c_2 * r_2 * (gbest_{j,g} - x_{j,g}^{(t)})\} \qquad (8)$$

$$¡ = \frac{2}{\left|2 - c - \sqrt{c^2 - 4c}\right|} \quad where\ c = c_1 + c_2,\ c < 4. \qquad (9)$$

4. Trapezoid membership function

From the purely mathematical viewpoint, we can have many different shapes of membership functions. In most practical applications, however, simple "trapezoid" membership functions work well, for which we use linear interpolation to get both endpoints of the interval [5].

Trapezoidal function: defined by a lower limit a, an upper limit d, a lower support limit b, and an upper support limit c, where a < b < c < d [7].



(10)

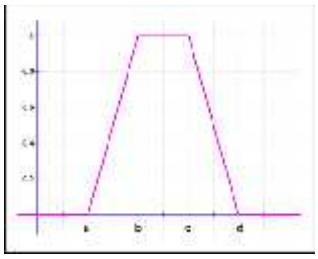
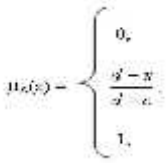


Fig. 1 Trapezidal function

There are two special cases of a trapezoidal function, which are called R-functions and L-functions:

R-functions: with parameters a = b = - ∞



(11)

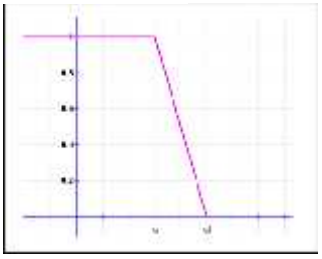
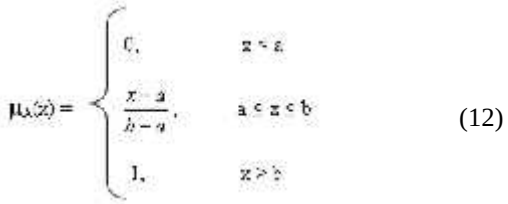


Fig. 2 Trapezidal R-function

L-Functions: with parameters c = d = + ∞



(12)

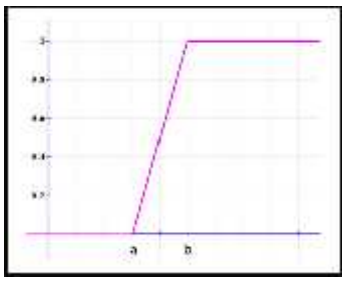


Fig. 3 Trapezidal L-function

5. Defuzzification method

One of defuzzification methods is Mean max membership: This method (also called middle-of-maxima) is closely related to the first method, except that the locations of the maximum membership can be non-unique (i.e., the maximum membership can be a plateau rather than a single point), as described in fig. 4. This method is given by the expression [13]

$$z^* = \frac{a+b}{2}. \qquad (13)$$

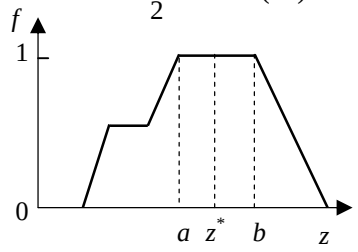


Fig. 4 Mean max membership defuzzification method

6. Solving fuzzy MSTP by PSO algorithm

Now, we will develop an algorithm to solve machine time scheduling problem where the starting time for each machine in each cycle is fuzzy with trapezoid membership function by using particle swarm optimization algorithm with construction factor. The new algorithm will be called PSO-FMTSP, follow the next steps:

1- Reformulation:

Each machine boundaries will be reformulate (calculate the new boundaries) based on its' successors machines boundaries. For each machine the new lower boundary called *h* and the new upper boundary called *H*.

2- Initial iteration:

First, the particle defines as a set of starting times for the machines in all cycles. The particle is represented by *D*-dimensional, where *D* equal to *N* multiplies by *K* (where *N* number of machine and *K* number of cycles). The x_{irpt} is the starting time for machine i in cycle r in particle p, $p = 1,2,..,Q$ in iteration t, $t = 1,2,..,T$ (where Q is number of particles in the swarm and T is number of iterations) which satisfy the constraints in eq. (5). $x_{irpt} \geq \max_{j \in N^{(i)}} (x_{j(r-1)pt} + p_j)$. Determine the $pbest_p$ which is the best position of particle p that make the best value of the objective function. Then determine the *gbest* which is the best particle that make the best value of the objective function in all iterations.

3- Other iterations:

The next iteration created by modifying the velocity of each particle by eq. (8), (9). Then the particle position will be update by the following eq. (7).

4- **Calculate** the objective function then find the $pbest_p$ and *gbest*
5- ***Repeat*** step 3 to step 5 until the *T*.

PSO-FMTSP algorithm:

Now, the PSO-MTSP algorithm will be used to deals with the fuzzy MTSP as follows:

PSO-FMTSP Algorithm:

A1: Reformulate the boundaries for each machine in each cycle as follows:

Put $H_{ik} = L_{ik} \quad \forall i \in N$, $h_{jr} = l_{jr} \quad \forall j \in N$,

$H_{jr} = \min(L_{jr}, (\min_{i \in U_j} H_{ir+1} - p_j))$,

where $U_j = \{ i \in N : j \in N^{(i)} \}$, $r = k-1,...,2,1$

A2: Put $t = 1$.
A3: Put $p = 1$.
A4: Put $r = 1$.
A5: Put $i = 1$.
A6: If $r \neq 1$ then $h_{ir} = \max_{j \in N^{(i)}} (x_{j(r-1)pt} + p_j)$.

A7: Generate random number for x_{irpt} where $h_{ir} \leq x_{irpt} \leq H_{ir}$.

A8: If $i < n$ then $i = i + 1$, go to A6.
A9: If $r < k$ then $r = r + 1$, go to A5.

A10: $pbest_p = f(x_{irpt})$, $\exists i = 1,.., N$, $\exists r = 1,.., K$.

A11: If $p < Q$ then $p = p + 1$, go to A4.
A12: find $\min(f(pbest_p))$, $\exists p = 1,.., Q$.

A13: $gbest = pbest_{p_{\min}}$.

A14: $t = t + 1$.
A15: Put $p = 1$.

A16: $i \{ v_{irpt} = w * v_{irp(t-1)} + c_1 * r_1 * (pbest_p - x_{irp(t-1)}) + c_2 * r_2 * (gbest - x_{irp(t-1)}) \}$.

$i = \dfrac{2}{\left|2 - c - \sqrt{c^2 - 4c}\right|}$ *where* $c = c_1 + c_2, \; c < 4.$

A17: $x_{irpt} = x_{irp(t-1)} + v_{irpt}$.

A18: if x_{irpt} is not feasible then go to A20.

A19: if $f(x_{irpt}) > f(x_{irpt} - 1)$ then $pbest_p = x_{irpt}$

A20: if $p < Q$ then $p = p + 1$, go to A16.
A21: if $f(gbest) > f(pbest_{p_{\min}})$ then $gbest = pbest_{p_{\min}}$.

A22: $f(gbest_t) < f(gbest_{t-1})$ then go to 15.

A23: if $f(gbest) > f(x_{ir1t})$ then $gbest = x_{ir1t}$.

A24: If $t < T$ then go to A15.
A25: The solution is *gbest.*

7. Numerical Example

Consider a problem with the following values of parameters n = 5 so $N = \{1,2,3,4,5\}$, and processing time $p = \{2,4.5,6.25,4,5\}$. The machines boundaries in each cycle in table 1 and the machines predecessor relation show in table 2 as follows:

Table 1: Machine Boundaries

Cycle (r)	*r = 1*	*r = 2*	*r = 3*
l_{ir} i=1,2,…,5	{1,0,0,3,1}	{4,6,6,5,6}	{10,11,12,9,11.5}
L_{ir} i=1,2,…,5	{5,4,3,5,6}	{6.5,7,7.5,7.25,6.5}	{13,12,15,12,14}

Table 2: Machine Relations

I	*1*	*2*	*3*	*4*	*5*
$N^{(i)}$	{1,2,3}	{2}	{2,3}	{1,4,5}	{1,3,5}
U_j	{1,4,5}	{1,2,3}	{1,3,5}	{4}	{4,5}

Assume further that
$f_{jr}(x_{jr}) = \max(a_{jr} - x_{jr}, x_{jr} + p_{jr} - b_{jr}, 0) \qquad \forall j \in N$

Where a_j, b_j are for all $j \in N$ given constants so that we have in our case for all $j \in N$

$f_{jr}^{(1)}(x_{jr}) = a_{jr} - x_{jr}$ $\qquad$ $f_{jr}^{(2)}(x_{jr} + p_{jr}) = x_{jr} + p_{jr} - b_{jr}$

Input values of a_{ir} and b_{ir} for each cycle

Table 3: Machines Penalty Boundaries

Cycle (r)	*r=1*	*r=2*	*r=3*
a_{ir} i=1,2,…,5	{1,1,1,3,3}	{5,7,6,5,7}	{11,12,11,10,13}
b_{ir} i=1,2,…,5	{4,6,8,5,5}	{8,9,8,6.5,8}	{13,15,14,12,14}

After Reformulation of the problem will obtain the following new boundary vectors:

Table 4: Reformulated Machine Boundaries

Cycle (r)	*r = 1*	*r = 2*	*r = 3*
h_{ir} i=1,2,…,5	{1,0,0,3,1}	{4,6,6,5,6}	{10,11,12,9,11.5}
H_{ir} i=1,2,…,5	{4.5,2,0.25,3.25,1.5}	{6.5,7,7.5,7.25,6.5}	{13,12,15,12,14}

We applied the HPSOM-SMTSP algorithm on this example. We found that, the best parameters for the

algorithm are swarm size equal 80, the value of w equal 0.5 and the value c1 and c2 equal 1.7. We run the program 100 trials.

After applying the previous algorithm on our example, we found that the shapes of memberships function for each machine in all cycle as follow as shown in fig. 5.

To find the value of starting time for each machine in each cycle, the membership function for each machine will be defuzzified. We will defuzzify the membership function by mean max method

The starting time for the each machine in each cycle is as in the following table 6:

Table 5: The crisp value for the fuzzy starting time variable

	M_1	M_2	M_3	M_4	M_5
C_1	1.7	1.65	0.05	3.1	1.2
C_2	6.2	6.1	6.2	6.8	6.3
C_3	12.49	11.3	12.49	11.3	12.49

8. Conclusion

An algorithm has been developed for solving machine time scheduling problem when starting time is fuzzy and has trapezoid membership function. particle swarm optimization with construction factor has been used to adapted the shape of the trapezoid membership. Then the mean max method has been used to find the crisp value for the fuzzy starting time for each machine.

References

[1] A. A. Sawy and A. A. Tharwat, " Comparison of Particle SWARM Optimization, Genetic Algorithm and Max separable Technique for Machine Time Scheduling Problem ", 5th international Conference on Mathematics and Engineering Physics, May 2010.

[2] A. Tharwat and A. Abuel-Yazid: "Generalized Algorithm For Muulti-Cycle Machine Time Scheduling", Proceeding of Meatip3 Conference, Assuit, Egypt, 2002.

[3] A. Tharwat and A. Abuel-Yazid: "Multi-Cycles Machine Time Scheduling Problem", First International Conference on Informatics and Systems, Cairo, Egypt, 2002.

[4] A. Tharwat and K. Zimmermann: "Optimal Choice of Parameters in Machine Time Scheduling Problems Case I," Conference MMEI, Liberc, Czech Republic, 1998.

[5] Aditi Barua, Lalitha Snigdha Mudunuri, and Olga Kosheleva "Why Trapezoidal and Triangular Membership Functions Work So Well: Towards a Theoretical Explanation"Journal of Uncertain Systems Vol.8, 2014

[6] Alec Banks, Jonathan Vincent, Chukwudi Anyakoha "A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications" NATURAL COMPUTING Volume 7, Number 1, 109-124.

[7] http://www.dma.fi.upm.es/java/fuzzy/fuzzyinf/funpert_en.htm: june 2014.

[8] M. Vlach and K. Zimmermann, "Machine Time Scheduling Synchronization of Starting Times", the Proceeding of the MME'99 Conference, Prague, Czech Republic, 1999.

[9] S. A. Hassan, A.A. Tharwat, I.A. El-Khodary, A. A. El-Sawy "Using Monte Carlo Simulation to Solve Machine Time Scheduling Problems With Stochastic Processing Time", Mathematical Methods in Economics conference: MME'2003, Prague , Czech Republic , 10-12 Sept.

[10] S. Panda and N. P. Padhy, "Comparison of Particle Swarm Optimization and Genetic Algorithm for TCSC-based Controller Design", International Journal of Computer Science and Engineering, Volume 1 Number 1, 2007

[11] S. Zlobec: "Input Optimization I: Optimal Realizations of Mathematical Models," Mathematical Programming, V 31, pp.245-268, 1985.

[12] S. Zlobec: "Input Optimization III: Optimal Realizations of Mathematical Models," Mathematical Programming, V 17, No 4, pp.429-445, 1986.

[13] T. J. Ross, "Fuzzy Logic with Engineering Application", Second Edition, John Wiley&Sons Ltd, 2004.

[14] Y. Sok and K. Zimmermann: "Optimal Choice of Parameters in Machine Time Scheduling Problems with Penalized Earliness in Starting Time and Lateness", AUC-Mathematica et Physica, V33, No 1, pp.53-61. 1992.

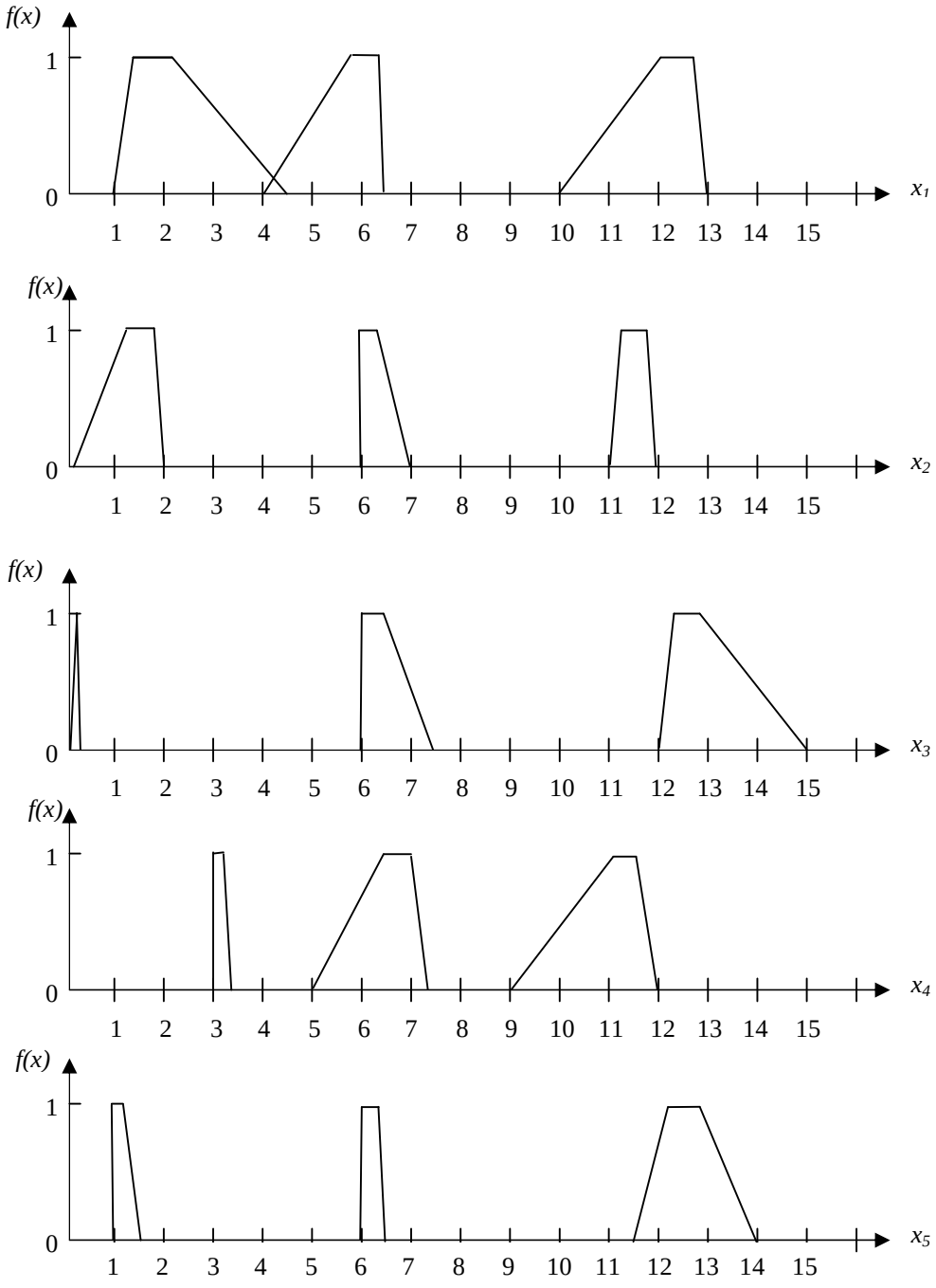


Fig. 5 The shape of membership function for x_i.