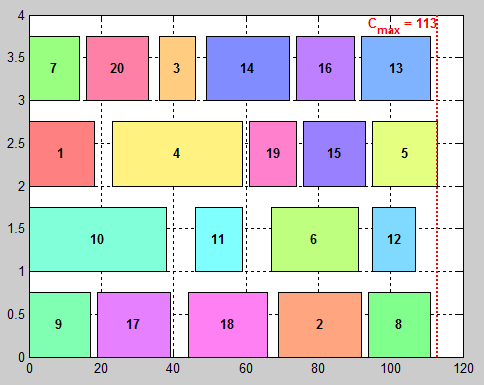
**حل مسئله زمانبندی ماشین ها موازی Parallel Machine Schedulingبا تبریدSA در متلب**

**J ماشین و I کار داريم. هر ماشين مي تواند تعدادي کار انجام دهد. فرض مي کنيم هر کار تنها توسط يک ماشين قابل انجام است. هر دو کار i و j سه وضعيت متفاوت نسبت به هم مي توانند داشته باشند: - i پيش نياز j است. به عبارت ديگر، j وقتي قابل اجرا است که i اجرا شده باشد. - j پيش نياز i است. - دو کار با هم ارتباطي ندارند. همچنين اجراي کار i، به زمان ti احتياج دارد. هر ماشين در هر لحظه تنها مي تواند يک کار را انجام دهد. هدف این است که کارها با توالی در ماشین ها اجرا شوند که Cmax (حداکثر زمانی که طول می کشد تا همه کارها در ماشین ها اجرا شوند) حداقل شود.**

****

**صورت مسئله:**

20 تا کار و 4 تا ماشین به شرح زیر داریم می خواهیم این کارها در این ماشین ها طوری اجرا شود که **حداکثر زمانی که طول می کشد تا همه کارها در ماشین ها اجرا شوندCmax، حداقل شود.**

**20 کار و 4 ماشین: 20x4**

p=[ 48 27 18 15

23 52 50 59

35 39 25 10

45 38 36 49

55 56 18 51

58 24 40 54

37 48 23 14

17 48 43 30

17 29 45 23

23 38 48 50

52 13 32 32

22 12 14 56

51 37 21 19

22 49 56 23

57 57 17 17

27 16 52 16

20 39 37 54

22 33 60 39

41 10 13 38

34 27 32 17]

وابستگی زمانبندی کارها در هر ماشین این صورت است:

یک ماتریس 20x20 برای 4 ماشین:

s(:,:,1)=[4 7 5 7 7 5 2 7 5 3 8 6 6 6 7 2 6 2 8 6

3 5 8 5 6 6 5 2 7 4 2 2 5 2 4 7 5 2 3 4

6 8 6 8 3 2 7 8 4 2 3 2 4 7 3 4 5 3 3 4

3 4 3 6 6 6 8 8 5 5 2 7 2 2 2 6 6 3 4 5

2 7 3 6 2 4 3 8 2 4 5 8 7 2 7 8 2 4 2 4

7 4 4 7 6 2 3 8 3 3 2 5 4 6 3 5 4 4 6 4

3 7 7 8 6 5 5 7 6 3 8 2 6 4 4 6 7 3 4 5

5 7 7 8 7 3 6 5 4 8 3 7 7 6 5 7 6 3 8 7

6 4 7 2 8 2 4 3 8 6 2 4 2 7 3 5 2 8 4 4

4 3 4 8 8 3 3 4 2 5 4 4 2 6 6 6 2 6 6 5

7 7 5 6 7 3 8 2 8 8 5 7 5 7 5 2 2 5 3 2

4 8 2 8 6 3 2 2 5 2 2 2 5 3 3 8 2 3 4 2

6 4 2 5 8 2 2 8 6 7 8 2 8 7 7 3 4 3 3 4

6 6 2 5 6 6 2 4 8 7 4 6 7 8 2 3 6 2 7 4

5 5 6 7 2 3 3 4 4 5 4 6 7 8 4 7 7 8 8 6

2 7 5 3 2 5 6 4 4 3 2 5 2 2 3 5 5 6 4 8

4 7 3 5 8 6 6 5 5 6 4 7 2 4 5 7 2 5 6 8

4 3 5 8 5 5 2 6 7 4 2 6 2 4 2 4 6 4 4 5

3 8 3 6 7 5 8 2 7 2 5 7 7 6 4 3 2 3 5 3

3 8 2 7 3 5 7 7 2 3 7 4 8 6 2 2 2 6 7 7]

s(:,:,2)=[7 7 7 6 3 3 2 4 7 2 5 7 3 5 4 4 5 8 4 5

7 7 3 4 4 3 3 6 6 3 5 4 3 5 2 2 6 5 6 3

7 2 2 8 2 5 3 7 2 2 8 5 6 8 3 3 4 7 8 8

2 5 7 3 6 3 2 6 7 5 7 8 6 4 3 7 2 6 7 7

6 4 6 6 3 7 2 5 8 3 5 5 6 5 4 7 5 2 5 8

5 5 7 6 2 8 6 6 7 8 8 4 6 8 3 8 4 5 7 3

3 4 6 4 7 2 8 5 2 2 2 6 2 2 4 6 7 6 4 6

2 4 4 2 4 5 4 2 4 2 4 4 4 8 2 2 7 5 8 6

7 3 4 2 6 2 4 7 6 5 8 7 5 3 8 8 6 4 8 2

3 3 7 4 4 7 8 8 7 7 8 4 3 6 2 7 2 8 8 4

3 2 4 3 6 8 8 4 3 4 6 5 7 6 8 4 2 7 4 3

6 8 7 7 2 2 6 8 3 3 6 6 7 6 4 5 5 7 5 7

8 6 7 4 8 8 8 4 6 4 4 8 3 4 2 8 4 4 3 3

5 8 7 7 7 2 7 8 5 3 8 4 7 6 4 7 8 6 7 8

6 3 5 7 7 6 4 5 6 5 2 7 2 7 7 7 8 8 8 7

3 8 6 5 7 7 6 4 3 8 7 7 7 2 7 5 4 8 8 4

8 7 8 3 4 5 3 3 3 6 6 8 2 2 5 5 7 6 5 5

5 6 5 8 6 8 4 2 7 2 7 2 6 8 6 5 8 3 6 6

6 5 2 3 6 8 6 4 7 4 4 4 4 6 8 3 6 6 3 7

2 3 8 8 5 6 5 7 8 2 7 6 7 3 2 7 8 2 8 6]

s(:,:,3)=[6 5 8 5 4 6 3 8 2 3 6 5 3 6 7 2 6 5 7 8

4 6 5 6 5 5 5 6 3 2 6 7 2 5 4 6 6 7 6 5

5 8 5 7 4 3 2 5 2 6 5 3 4 6 6 2 3 8 8 2

6 7 4 5 7 6 7 7 5 8 3 4 6 3 2 6 2 7 2 2

8 4 5 3 7 2 7 5 3 8 7 3 6 2 2 7 3 4 6 7

7 7 5 5 5 6 8 5 4 3 3 4 5 5 8 3 8 5 3 5

2 2 2 4 6 6 8 6 4 5 4 4 5 3 3 5 8 7 7 4

6 2 8 8 8 2 5 4 2 4 8 5 4 8 6 5 6 2 3 7

5 2 2 6 7 2 3 3 5 5 7 2 5 8 8 2 7 2 5 4

5 3 5 6 6 3 2 6 6 3 4 5 7 4 3 5 3 3 4 5

2 4 7 7 2 2 5 8 3 2 4 3 7 2 3 6 6 5 7 6

7 4 4 4 4 5 6 4 7 5 6 3 6 6 4 3 7 8 6 8

4 2 6 5 6 7 7 2 2 3 8 3 7 7 8 7 4 6 3 4

3 5 7 5 5 6 2 5 4 2 8 3 6 8 4 8 8 4 4 6

4 2 8 3 2 5 6 4 2 8 6 8 2 2 3 7 2 4 2 8

4 3 8 5 3 8 5 4 3 5 4 8 5 5 3 5 4 7 6 2

5 6 3 6 7 2 3 7 2 8 7 7 4 6 4 3 5 8 5 6

5 8 3 4 2 8 8 4 3 7 5 7 2 6 4 7 2 6 3 4

4 8 8 7 8 2 6 4 2 2 8 3 3 7 2 3 7 3 3 4

4 5 6 7 2 5 5 4 3 6 2 4 3 6 5 8 5 2 5 3]

s(:,:,4)=[7 7 8 3 8 2 5 2 3 8 2 5 7 7 3 4 7 6 8 7

8 5 2 3 6 7 6 4 7 6 4 8 5 8 8 4 7 3 5 6

3 3 2 4 4 4 8 8 4 6 7 7 4 3 6 8 4 5 8 5

7 5 4 8 7 7 3 5 4 7 3 8 7 2 8 6 5 7 7 3

3 5 6 5 8 5 7 4 3 2 7 3 5 3 5 8 8 3 5 8

8 8 5 4 5 5 6 3 7 8 6 5 8 4 8 3 6 4 6 5

7 7 8 3 5 2 5 5 6 4 7 2 8 4 2 7 7 5 8 2

4 8 5 8 4 2 8 8 7 2 7 7 4 8 6 6 3 4 3 6

7 6 5 4 2 2 4 2 7 7 4 6 5 2 7 3 6 7 4 5

5 4 5 7 3 6 3 5 2 3 4 8 4 6 3 5 6 8 8 2

7 8 6 6 2 3 6 7 8 3 5 8 6 3 8 4 8 3 4 8

4 5 2 4 5 7 6 2 5 6 4 8 7 7 7 6 2 3 6 4

2 3 7 8 2 8 4 6 7 3 7 4 7 3 7 7 5 6 8 3

6 4 2 7 8 8 7 8 7 4 7 2 2 5 6 2 5 4 8 2

8 6 5 5 6 5 8 3 7 4 5 5 7 8 7 7 2 8 6 4

3 5 3 7 2 3 8 2 3 4 3 3 2 4 4 7 8 8 2 3

5 7 4 8 2 3 2 6 5 4 6 3 4 2 3 4 8 6 2 6

7 8 6 5 3 5 3 8 6 6 3 4 7 3 4 5 5 8 6 2

2 8 3 4 5 7 2 6 8 3 5 2 7 4 6 6 7 4 5 3

8 5 3 6 2 4 6 8 7 3 4 7 4 4 7 6 3 6 8 3]

**شرح کد:**

این سورس کد شامل 7 فایل می باشد که عبارتند از:

**:CreateModel.m** برای ایجاد مدل،کارها و ماشین ها و وابستگی های زمانی و کارها و مقدار دهی اولیه پارامتر های مدل مسیله از آن استفاده می شود.

function model=CreateModel()

p=[ 48 27 18 15

23 52 50 59

35 39 25 10

45 38 36 49

55 56 18 51

58 24 40 54

37 48 23 14

17 48 43 30

17 29 45 23

23 38 48 50

52 13 32 32

22 12 14 56

51 37 21 19

22 49 56 23

57 57 17 17

27 16 52 16

20 39 37 54

22 33 60 39

41 10 13 38

34 27 32 17];

I=size(p,1);

J=size(p,2);

s(:,:,1)=[4 7 5 7 7 5 2 7 5 3 8 6 6 6 7 2 6 2 8 6

3 5 8 5 6 6 5 2 7 4 2 2 5 2 4 7 5 2 3 4

6 8 6 8 3 2 7 8 4 2 3 2 4 7 3 4 5 3 3 4

3 4 3 6 6 6 8 8 5 5 2 7 2 2 2 6 6 3 4 5

2 7 3 6 2 4 3 8 2 4 5 8 7 2 7 8 2 4 2 4

7 4 4 7 6 2 3 8 3 3 2 5 4 6 3 5 4 4 6 4

3 7 7 8 6 5 5 7 6 3 8 2 6 4 4 6 7 3 4 5

5 7 7 8 7 3 6 5 4 8 3 7 7 6 5 7 6 3 8 7

6 4 7 2 8 2 4 3 8 6 2 4 2 7 3 5 2 8 4 4

4 3 4 8 8 3 3 4 2 5 4 4 2 6 6 6 2 6 6 5

7 7 5 6 7 3 8 2 8 8 5 7 5 7 5 2 2 5 3 2

4 8 2 8 6 3 2 2 5 2 2 2 5 3 3 8 2 3 4 2

6 4 2 5 8 2 2 8 6 7 8 2 8 7 7 3 4 3 3 4

6 6 2 5 6 6 2 4 8 7 4 6 7 8 2 3 6 2 7 4

5 5 6 7 2 3 3 4 4 5 4 6 7 8 4 7 7 8 8 6

2 7 5 3 2 5 6 4 4 3 2 5 2 2 3 5 5 6 4 8

4 7 3 5 8 6 6 5 5 6 4 7 2 4 5 7 2 5 6 8

4 3 5 8 5 5 2 6 7 4 2 6 2 4 2 4 6 4 4 5

3 8 3 6 7 5 8 2 7 2 5 7 7 6 4 3 2 3 5 3

3 8 2 7 3 5 7 7 2 3 7 4 8 6 2 2 2 6 7 7];

s(:,:,2)=[7 7 7 6 3 3 2 4 7 2 5 7 3 5 4 4 5 8 4 5

7 7 3 4 4 3 3 6 6 3 5 4 3 5 2 2 6 5 6 3

7 2 2 8 2 5 3 7 2 2 8 5 6 8 3 3 4 7 8 8

2 5 7 3 6 3 2 6 7 5 7 8 6 4 3 7 2 6 7 7

6 4 6 6 3 7 2 5 8 3 5 5 6 5 4 7 5 2 5 8

5 5 7 6 2 8 6 6 7 8 8 4 6 8 3 8 4 5 7 3

3 4 6 4 7 2 8 5 2 2 2 6 2 2 4 6 7 6 4 6

2 4 4 2 4 5 4 2 4 2 4 4 4 8 2 2 7 5 8 6

7 3 4 2 6 2 4 7 6 5 8 7 5 3 8 8 6 4 8 2

3 3 7 4 4 7 8 8 7 7 8 4 3 6 2 7 2 8 8 4

3 2 4 3 6 8 8 4 3 4 6 5 7 6 8 4 2 7 4 3

6 8 7 7 2 2 6 8 3 3 6 6 7 6 4 5 5 7 5 7

8 6 7 4 8 8 8 4 6 4 4 8 3 4 2 8 4 4 3 3

5 8 7 7 7 2 7 8 5 3 8 4 7 6 4 7 8 6 7 8

6 3 5 7 7 6 4 5 6 5 2 7 2 7 7 7 8 8 8 7

3 8 6 5 7 7 6 4 3 8 7 7 7 2 7 5 4 8 8 4

8 7 8 3 4 5 3 3 3 6 6 8 2 2 5 5 7 6 5 5

5 6 5 8 6 8 4 2 7 2 7 2 6 8 6 5 8 3 6 6

6 5 2 3 6 8 6 4 7 4 4 4 4 6 8 3 6 6 3 7

2 3 8 8 5 6 5 7 8 2 7 6 7 3 2 7 8 2 8 6];

s(:,:,3)=[6 5 8 5 4 6 3 8 2 3 6 5 3 6 7 2 6 5 7 8

4 6 5 6 5 5 5 6 3 2 6 7 2 5 4 6 6 7 6 5

5 8 5 7 4 3 2 5 2 6 5 3 4 6 6 2 3 8 8 2

6 7 4 5 7 6 7 7 5 8 3 4 6 3 2 6 2 7 2 2

8 4 5 3 7 2 7 5 3 8 7 3 6 2 2 7 3 4 6 7

7 7 5 5 5 6 8 5 4 3 3 4 5 5 8 3 8 5 3 5

2 2 2 4 6 6 8 6 4 5 4 4 5 3 3 5 8 7 7 4

6 2 8 8 8 2 5 4 2 4 8 5 4 8 6 5 6 2 3 7

5 2 2 6 7 2 3 3 5 5 7 2 5 8 8 2 7 2 5 4

5 3 5 6 6 3 2 6 6 3 4 5 7 4 3 5 3 3 4 5

2 4 7 7 2 2 5 8 3 2 4 3 7 2 3 6 6 5 7 6

7 4 4 4 4 5 6 4 7 5 6 3 6 6 4 3 7 8 6 8

4 2 6 5 6 7 7 2 2 3 8 3 7 7 8 7 4 6 3 4

3 5 7 5 5 6 2 5 4 2 8 3 6 8 4 8 8 4 4 6

4 2 8 3 2 5 6 4 2 8 6 8 2 2 3 7 2 4 2 8

4 3 8 5 3 8 5 4 3 5 4 8 5 5 3 5 4 7 6 2

5 6 3 6 7 2 3 7 2 8 7 7 4 6 4 3 5 8 5 6

5 8 3 4 2 8 8 4 3 7 5 7 2 6 4 7 2 6 3 4

4 8 8 7 8 2 6 4 2 2 8 3 3 7 2 3 7 3 3 4

4 5 6 7 2 5 5 4 3 6 2 4 3 6 5 8 5 2 5 3];

s(:,:,4)=[7 7 8 3 8 2 5 2 3 8 2 5 7 7 3 4 7 6 8 7

8 5 2 3 6 7 6 4 7 6 4 8 5 8 8 4 7 3 5 6

3 3 2 4 4 4 8 8 4 6 7 7 4 3 6 8 4 5 8 5

7 5 4 8 7 7 3 5 4 7 3 8 7 2 8 6 5 7 7 3

3 5 6 5 8 5 7 4 3 2 7 3 5 3 5 8 8 3 5 8

8 8 5 4 5 5 6 3 7 8 6 5 8 4 8 3 6 4 6 5

7 7 8 3 5 2 5 5 6 4 7 2 8 4 2 7 7 5 8 2

4 8 5 8 4 2 8 8 7 2 7 7 4 8 6 6 3 4 3 6

7 6 5 4 2 2 4 2 7 7 4 6 5 2 7 3 6 7 4 5

5 4 5 7 3 6 3 5 2 3 4 8 4 6 3 5 6 8 8 2

7 8 6 6 2 3 6 7 8 3 5 8 6 3 8 4 8 3 4 8

4 5 2 4 5 7 6 2 5 6 4 8 7 7 7 6 2 3 6 4

2 3 7 8 2 8 4 6 7 3 7 4 7 3 7 7 5 6 8 3

6 4 2 7 8 8 7 8 7 4 7 2 2 5 6 2 5 4 8 2

8 6 5 5 6 5 8 3 7 4 5 5 7 8 7 7 2 8 6 4

3 5 3 7 2 3 8 2 3 4 3 3 2 4 4 7 8 8 2 3

5 7 4 8 2 3 2 6 5 4 6 3 4 2 3 4 8 6 2 6

7 8 6 5 3 5 3 8 6 6 3 4 7 3 4 5 5 8 6 2

2 8 3 4 5 7 2 6 8 3 5 2 7 4 6 6 7 4 5 3

8 5 3 6 2 4 6 8 7 3 4 7 4 4 7 6 3 6 8 3];

model.I=I;

کار

model.J=J;

ماشین

model.p=p;

ماتریس شامل کار و ماشین

model.s=s;

زمانبندی هر ماشین

model.nVar=I+J-1;

تعداد متغیرها**21,22,23 جدا کننده هستند بین ماشین ها**

end

**MyCost.m:**این تابع برای مشخص کردن میزان هزینه راه حل می باشد. چون تبرید min یاب است هزینه برای ما مهم است.

function [z, sol]=MyCost(q,model)

اجرای تابع ParseSolution با مدل و q (جدا کننده کارها)

sol=ParseSolution(q,model);

زمانی که طول می کشد تا همه کارها در همه ماشین ها اجرا شوند.

z=sol.Cmax;

end

**:ParseSolution.m**راه حل را برای ما ایجاد می کند.

function sol=ParseSolution(q,model)

I=model.I;

J=model.J;

p=model.p;

s=model.s;

موقعیت جداکننده ها

DelPos=find(q>I);

ترتیب و شوع و پایان کارها را تعیین می کند.

From=[0 DelPos]+1;

To=[DelPos I+J]-1;

ایجاد لیست کارها

L=cell(J,1);

for j=1:J

L{j}=q(From(j):To(j));

end

شبیه سازی مبتنی بر زمان

ST=zeros(I,1);

PT=zeros(I,1);

FT=zeros(I,1);

MCT=zeros(J,1);

for j=1:J

for i=L{j}

k=find(L{j}==i);

if k==1

ST(i)=0;

else

PreviousJob=L{j}(k-1);

ST(i)=FT(PreviousJob)+s(PreviousJob,i,j);

end

PT(i)=p(i,j);

FT(i)=ST(i)+PT(i);

end

if ~isempty(L{j})

MCT(j)=FT(L{j}(end));

end

end

Cmax=max(MCT);

sol.L=L;

sol.ST=ST;

sol.PT=PT;

sol.FT=FT;

sol.MCT=MCT;

sol.Cmax=Cmax;

end

**:CreateRandomSolution.**mایجاد راه حل به صورت تصادفی

function q=CreateRandomSolution(model)

تعداد متغیرها

nVar=model.nVar;

یک جایگشت تصادفی از اعداد 1 تا nVar را بر می گرداند.

q=randperm(nVar);

end

**CreateNeighbor.m: ا**یجاد همسایه

function qnew=CreateNeighbor(q)

یک عدد تصادفی بین 1 تا3 تولید می کند.

m=randi([1 3]);

switch m

اگر عدد تولید شده 1 بود جابه جایی

case 1

جابه جایی

qnew=Swap(q);

اگر عدد تولید شده 2 بود معکوس

case 2

معکوس

qnew=Reversion(q);

اگر عدد تولید شده 3 بود اضافه کن

case 3

اضافه کن

qnew=Insertion(q);

end

end

function qnew=Swap(q)

n=numel(q);

i=randsample(n,2);

i1=i(1);

i2=i(2);

qnew=q;

qnew([i1 i2])=q([i2 i1]);

end

function qnew=Reversion(q)

n=numel(q);

i=randsample(n,2);

i1=min(i(1),i(2));

i2=max(i(1),i(2));

qnew=q;

qnew(i1:i2)=q(i2:-1:i1);

end

function qnew=Insertion(q)

n=numel(q);

i=randsample(n,2);

i1=i(1);

i2=i(2);

if i1<i2

qnew=[q(1:i1-1) q(i1+1:i2) q(i1) q(i2+1:end)];

else

qnew=[q(1:i2) q(i1) q(i2+1:i1-1) q(i1+1:end)];

end

end

**sa.m:** این فایل شامل کد الگوریتم تبرید به همراه توابع بالا برای پیاده سازی مسئله ست.

clc;

clear;

close all;

تعریف مسیله

model=CreateModel(); ایجاد مدل مسیله

CostFunction=@(q) MyCost(q,model); تابع هزینه

nVar=model.nVar;

(کارها به همراه جداکننده برای هر ماشین)تعداد متغیرهای تصمیم

VarSize=[1 nVar]; اندازه ماتریس متغیرها

پارامترهای تبرید

MaxIt=500; ماکزیمم تعداد تکرار

MaxIt2=25; ماکزیمم تعداد تکرار داخلی

T0=10; دمای اولیه

alpha=0.97; نرخ میرایی دما

مقداردهی اولیه

ایجاد راه حل اولیه

x.Position=CreateRandomSolution(model);

[x.Cost, x.Sol]=CostFunction(x.Position);

بروزرسانی بهترین راه حل که تاکنون پیدا شده است

BestSol=x;

آرایه برای نگه داشتن بهترین مقادیر هزینه

BestCost=zeros(MaxIt,1);

تنظیم دمای اولیه

T=T0;

حلقه اصلی تبرید

for it=1:MaxIt

for it2=1:MaxIt2

ایجاد همسایه

xnew.Position=CreateNeighbor(x.Position);

[xnew.Cost, xnew.Sol]=CostFunction(xnew.Position);

if xnew.Cost<=x.Cost

% xnew بهتر است ، بنابراین پذیرفته می شود

x=xnew;

else

% xnew بهتر نیست ، بنابراین به طور مشروط پذیرفته می شود delta=xnew.Cost-x.Cost;

p=exp(-delta/T);

if rand<=p

x=xnew;

end

end

آپدیت بهترن راه حل

if x.Cost<=BestSol.Cost

BestSol=x;

end

end

بهترین هزینه را ذخیره کنید

BestCost(it)=BestSol.Cost;

نمایش اطلاعات تکرار

disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCost(it))]);

کاهش دما

T=alpha\*T;

کشیدن راه حل

figure(1);

PlotSolution(BestSol.Sol,model);

pause(0.01);

end

نتایج

figure;

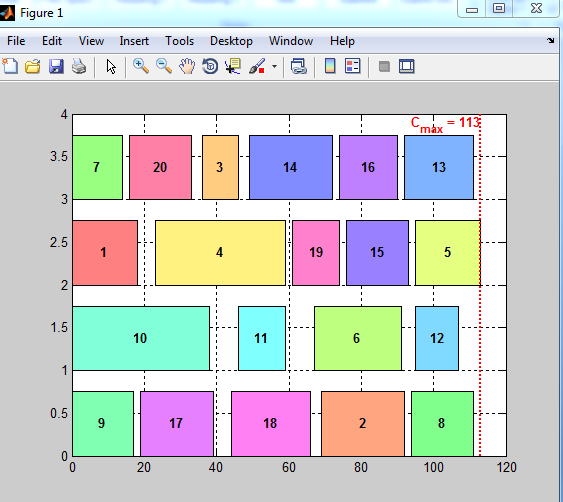
plot(BestCost,'LineWidth',2);

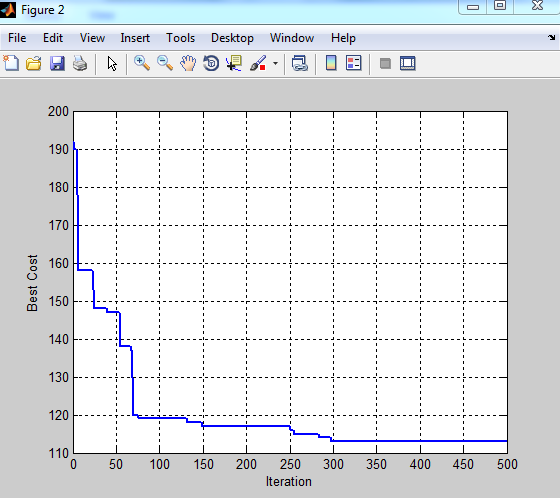
xlabel('Iteration');

ylabel('Best Cost');

grid on;

**نتایج:**

****

****

**Iteration 1: Best Cost = 192**

**Iteration 2: Best Cost = 190**

**Iteration 3: Best Cost = 190**

**Iteration 4: Best Cost = 190**

**Iteration 5: Best Cost = 178**

**Iteration 6: Best Cost = 158**

**Iteration 7: Best Cost = 158**

**Iteration 8: Best Cost = 158**

**Iteration 9: Best Cost = 158**

**Iteration 10: Best Cost = 158**

**Iteration 11: Best Cost = 158**

**Iteration 12: Best Cost = 158**

**Iteration 13: Best Cost = 158**

**Iteration 14: Best Cost = 158**

**Iteration 15: Best Cost = 158**

**Iteration 16: Best Cost = 158**

**Iteration 17: Best Cost = 158**

**Iteration 18: Best Cost = 158**

**Iteration 19: Best Cost = 158**

**Iteration 20: Best Cost = 158**

**Iteration 21: Best Cost = 158**

**Iteration 22: Best Cost = 158**

**Iteration 23: Best Cost = 158**

**Iteration 24: Best Cost = 153**

**Iteration 25: Best Cost = 148**

**Iteration 26: Best Cost = 148**

**Iteration 27: Best Cost = 148**

**Iteration 28: Best Cost = 148**

**Iteration 29: Best Cost = 148**

**Iteration 30: Best Cost = 148**

**Iteration 31: Best Cost = 148**

**Iteration 32: Best Cost = 148**

**Iteration 33: Best Cost = 148**

**Iteration 34: Best Cost = 148**

**Iteration 35: Best Cost = 148**

**Iteration 36: Best Cost = 148**

**Iteration 37: Best Cost = 148**

**Iteration 38: Best Cost = 148**

**Iteration 39: Best Cost = 148**

**Iteration 40: Best Cost = 147**

**Iteration 41: Best Cost = 147**

**Iteration 42: Best Cost = 147**

**Iteration 43: Best Cost = 147**

**Iteration 44: Best Cost = 147**

**Iteration 45: Best Cost = 147**

**Iteration 46: Best Cost = 147**

**Iteration 47: Best Cost = 147**

**Iteration 48: Best Cost = 147**

**Iteration 49: Best Cost = 147**

**Iteration 50: Best Cost = 147**

**Iteration 51: Best Cost = 147**

**Iteration 52: Best Cost = 147**

**Iteration 53: Best Cost = 147**

**Iteration 54: Best Cost = 147**

**Iteration 55: Best Cost = 138**

**Iteration 56: Best Cost = 138**

**Iteration 57: Best Cost = 138**

**Iteration 58: Best Cost = 138**

**Iteration 59: Best Cost = 138**

**Iteration 60: Best Cost = 138**

**Iteration 61: Best Cost = 138**

**Iteration 62: Best Cost = 138**

**Iteration 63: Best Cost = 138**

**Iteration 64: Best Cost = 138**

**Iteration 65: Best Cost = 138**

**Iteration 66: Best Cost = 138**

**Iteration 67: Best Cost = 137**

**Iteration 68: Best Cost = 137**

**Iteration 69: Best Cost = 130**

**Iteration 70: Best Cost = 120**

**Iteration 71: Best Cost = 120**

**Iteration 72: Best Cost = 120**

**Iteration 73: Best Cost = 120**

**Iteration 74: Best Cost = 120**

**Iteration 75: Best Cost = 120**

**Iteration 76: Best Cost = 119**

**Iteration 77: Best Cost = 119**

**Iteration 78: Best Cost = 119**

**Iteration 79: Best Cost = 119**

**Iteration 80: Best Cost = 119**

**Iteration 81: Best Cost = 119**

**Iteration 82: Best Cost = 119**

**Iteration 83: Best Cost = 119**

**Iteration 84: Best Cost = 119**

**Iteration 85: Best Cost = 119**

**Iteration 86: Best Cost = 119**

**Iteration 87: Best Cost = 119**

**Iteration 88: Best Cost = 119**

**Iteration 89: Best Cost = 119**

**Iteration 90: Best Cost = 119**

**Iteration 91: Best Cost = 119**

**Iteration 92: Best Cost = 119**

**Iteration 93: Best Cost = 119**

**Iteration 94: Best Cost = 119**

**Iteration 95: Best Cost = 119**

**Iteration 96: Best Cost = 119**

**Iteration 97: Best Cost = 119**

**Iteration 98: Best Cost = 119**

**Iteration 99: Best Cost = 119**

**Iteration 100: Best Cost = 119**

**Iteration 101: Best Cost = 119**

**Iteration 102: Best Cost = 119**

**Iteration 103: Best Cost = 119**

**Iteration 104: Best Cost = 119**

**Iteration 105: Best Cost = 119**

**Iteration 106: Best Cost = 119**

**Iteration 107: Best Cost = 119**

**Iteration 108: Best Cost = 119**

**Iteration 109: Best Cost = 119**

**Iteration 110: Best Cost = 119**

**Iteration 111: Best Cost = 119**

**Iteration 112: Best Cost = 119**

**Iteration 113: Best Cost = 119**

**Iteration 114: Best Cost = 119**

**Iteration 115: Best Cost = 119**

**Iteration 116: Best Cost = 119**

**Iteration 117: Best Cost = 119**

**Iteration 118: Best Cost = 119**

**Iteration 119: Best Cost = 119**

**Iteration 120: Best Cost = 119**

**Iteration 121: Best Cost = 119**

**Iteration 122: Best Cost = 119**

**Iteration 123: Best Cost = 119**

**Iteration 124: Best Cost = 119**

**Iteration 125: Best Cost = 119**

**Iteration 126: Best Cost = 119**

**Iteration 127: Best Cost = 119**

**Iteration 128: Best Cost = 119**

**Iteration 129: Best Cost = 119**

**Iteration 130: Best Cost = 119**

**Iteration 131: Best Cost = 119**

**Iteration 132: Best Cost = 118**

**Iteration 133: Best Cost = 118**

**Iteration 134: Best Cost = 118**

**Iteration 135: Best Cost = 118**

**Iteration 136: Best Cost = 118**

**Iteration 137: Best Cost = 118**

**Iteration 138: Best Cost = 118**

**Iteration 139: Best Cost = 118**

**Iteration 140: Best Cost = 118**

**Iteration 141: Best Cost = 118**

**Iteration 142: Best Cost = 118**

**Iteration 143: Best Cost = 118**

**Iteration 144: Best Cost = 118**

**Iteration 145: Best Cost = 118**

**Iteration 146: Best Cost = 118**

**Iteration 147: Best Cost = 118**

**Iteration 148: Best Cost = 118**

**Iteration 149: Best Cost = 117**

**Iteration 150: Best Cost = 117**

**Iteration 151: Best Cost = 117**

**Iteration 152: Best Cost = 117**

**Iteration 153: Best Cost = 117**

**Iteration 154: Best Cost = 117**

**Iteration 155: Best Cost = 117**

**Iteration 156: Best Cost = 117**

**Iteration 157: Best Cost = 117**

**Iteration 158: Best Cost = 117**

**Iteration 159: Best Cost = 117**

**Iteration 160: Best Cost = 117**

**Iteration 161: Best Cost = 117**

**Iteration 162: Best Cost = 117**

**Iteration 163: Best Cost = 117**

**Iteration 164: Best Cost = 117**

**Iteration 165: Best Cost = 117**

**Iteration 166: Best Cost = 117**

**Iteration 167: Best Cost = 117**

**Iteration 168: Best Cost = 117**

**Iteration 169: Best Cost = 117**

**Iteration 170: Best Cost = 117**

**Iteration 171: Best Cost = 117**

**Iteration 172: Best Cost = 117**

**Iteration 173: Best Cost = 117**

**Iteration 174: Best Cost = 117**

**Iteration 175: Best Cost = 117**

**Iteration 176: Best Cost = 117**

**Iteration 177: Best Cost = 117**

**Iteration 178: Best Cost = 117**

**Iteration 179: Best Cost = 117**

**Iteration 180: Best Cost = 117**

**Iteration 181: Best Cost = 117**

**Iteration 182: Best Cost = 117**

**Iteration 183: Best Cost = 117**

**Iteration 184: Best Cost = 117**

**Iteration 185: Best Cost = 117**

**Iteration 186: Best Cost = 117**

**Iteration 187: Best Cost = 117**

**Iteration 188: Best Cost = 117**

**Iteration 189: Best Cost = 117**

**Iteration 190: Best Cost = 117**

**Iteration 191: Best Cost = 117**

**Iteration 192: Best Cost = 117**

**Iteration 193: Best Cost = 117**

**Iteration 194: Best Cost = 117**

**Iteration 195: Best Cost = 117**

**Iteration 196: Best Cost = 117**

**Iteration 197: Best Cost = 117**

**Iteration 198: Best Cost = 117**

**Iteration 199: Best Cost = 117**

**Iteration 200: Best Cost = 117**

**Iteration 201: Best Cost = 117**

**Iteration 202: Best Cost = 117**

**Iteration 203: Best Cost = 117**

**Iteration 204: Best Cost = 117**

**Iteration 205: Best Cost = 117**

**Iteration 206: Best Cost = 117**

**Iteration 207: Best Cost = 117**

**Iteration 208: Best Cost = 117**

**Iteration 209: Best Cost = 117**

**Iteration 210: Best Cost = 117**

**Iteration 211: Best Cost = 117**

**Iteration 212: Best Cost = 117**

**Iteration 213: Best Cost = 117**

**Iteration 214: Best Cost = 117**

**Iteration 215: Best Cost = 117**

**Iteration 216: Best Cost = 117**

**Iteration 217: Best Cost = 117**

**Iteration 218: Best Cost = 117**

**Iteration 219: Best Cost = 117**

**Iteration 220: Best Cost = 117**

**Iteration 221: Best Cost = 117**

**Iteration 222: Best Cost = 117**

**Iteration 223: Best Cost = 117**

**Iteration 224: Best Cost = 117**

**Iteration 225: Best Cost = 117**

**Iteration 226: Best Cost = 117**

**Iteration 227: Best Cost = 117**

**Iteration 228: Best Cost = 117**

**Iteration 229: Best Cost = 117**

**Iteration 230: Best Cost = 117**

**Iteration 231: Best Cost = 117**

**Iteration 232: Best Cost = 117**

**Iteration 233: Best Cost = 117**

**Iteration 234: Best Cost = 117**

**Iteration 235: Best Cost = 117**

**Iteration 236: Best Cost = 117**

**Iteration 237: Best Cost = 117**

**Iteration 238: Best Cost = 117**

**Iteration 239: Best Cost = 117**

**Iteration 240: Best Cost = 117**

**Iteration 241: Best Cost = 117**

**Iteration 242: Best Cost = 117**

**Iteration 243: Best Cost = 117**

**Iteration 244: Best Cost = 117**

**Iteration 245: Best Cost = 117**

**Iteration 246: Best Cost = 117**

**Iteration 247: Best Cost = 117**

**Iteration 248: Best Cost = 117**

**Iteration 249: Best Cost = 117**

**Iteration 250: Best Cost = 117**

**Iteration 251: Best Cost = 116**

**Iteration 252: Best Cost = 116**

**Iteration 253: Best Cost = 116**

**Iteration 254: Best Cost = 116**

**Iteration 255: Best Cost = 115**

**Iteration 256: Best Cost = 115**

**Iteration 257: Best Cost = 115**

**Iteration 258: Best Cost = 115**

**Iteration 259: Best Cost = 115**

**Iteration 260: Best Cost = 115**

**Iteration 261: Best Cost = 115**

**Iteration 262: Best Cost = 115**

**Iteration 263: Best Cost = 115**

**Iteration 264: Best Cost = 115**

**Iteration 265: Best Cost = 115**

**Iteration 266: Best Cost = 115**

**Iteration 267: Best Cost = 115**

**Iteration 268: Best Cost = 115**

**Iteration 269: Best Cost = 115**

**Iteration 270: Best Cost = 115**

**Iteration 271: Best Cost = 115**

**Iteration 272: Best Cost = 115**

**Iteration 273: Best Cost = 115**

**Iteration 274: Best Cost = 115**

**Iteration 275: Best Cost = 115**

**Iteration 276: Best Cost = 115**

**Iteration 277: Best Cost = 115**

**Iteration 278: Best Cost = 115**

**Iteration 279: Best Cost = 115**

**Iteration 280: Best Cost = 115**

**Iteration 281: Best Cost = 115**

**Iteration 282: Best Cost = 115**

**Iteration 283: Best Cost = 115**

**Iteration 284: Best Cost = 114**

**Iteration 285: Best Cost = 114**

**Iteration 286: Best Cost = 114**

**Iteration 287: Best Cost = 114**

**Iteration 288: Best Cost = 114**

**Iteration 289: Best Cost = 114**

**Iteration 290: Best Cost = 114**

**Iteration 291: Best Cost = 114**

**Iteration 292: Best Cost = 114**

**Iteration 293: Best Cost = 114**

**Iteration 294: Best Cost = 114**

**Iteration 295: Best Cost = 114**

**Iteration 296: Best Cost = 114**

**Iteration 297: Best Cost = 114**

**Iteration 298: Best Cost = 113**

**Iteration 299: Best Cost = 113**

**Iteration 300: Best Cost = 113**

**Iteration 301: Best Cost = 113**

**Iteration 302: Best Cost = 113**

**Iteration 303: Best Cost = 113**

**Iteration 304: Best Cost = 113**

**Iteration 305: Best Cost = 113**

**Iteration 306: Best Cost = 113**

**Iteration 307: Best Cost = 113**

**Iteration 308: Best Cost = 113**

**Iteration 309: Best Cost = 113**

**Iteration 310: Best Cost = 113**

**Iteration 311: Best Cost = 113**

**Iteration 312: Best Cost = 113**

**Iteration 313: Best Cost = 113**

**Iteration 314: Best Cost = 113**

**Iteration 315: Best Cost = 113**

**Iteration 316: Best Cost = 113**

**Iteration 317: Best Cost = 113**

**Iteration 318: Best Cost = 113**

**Iteration 319: Best Cost = 113**

**Iteration 320: Best Cost = 113**

**Iteration 321: Best Cost = 113**

**Iteration 322: Best Cost = 113**

**Iteration 323: Best Cost = 113**

**Iteration 324: Best Cost = 113**

**Iteration 325: Best Cost = 113**

**Iteration 326: Best Cost = 113**

**Iteration 327: Best Cost = 113**

**Iteration 328: Best Cost = 113**

**Iteration 329: Best Cost = 113**

**Iteration 330: Best Cost = 113**

**Iteration 331: Best Cost = 113**

**Iteration 332: Best Cost = 113**

**Iteration 333: Best Cost = 113**

**Iteration 334: Best Cost = 113**

**Iteration 335: Best Cost = 113**

**Iteration 336: Best Cost = 113**

**Iteration 337: Best Cost = 113**

**Iteration 338: Best Cost = 113**

**Iteration 339: Best Cost = 113**

**Iteration 340: Best Cost = 113**

**Iteration 341: Best Cost = 113**

**Iteration 342: Best Cost = 113**

**Iteration 343: Best Cost = 113**

**Iteration 344: Best Cost = 113**

**Iteration 345: Best Cost = 113**

**Iteration 346: Best Cost = 113**

**Iteration 347: Best Cost = 113**

**Iteration 348: Best Cost = 113**

**Iteration 349: Best Cost = 113**

**Iteration 350: Best Cost = 113**

**Iteration 351: Best Cost = 113**

**Iteration 352: Best Cost = 113**

**Iteration 353: Best Cost = 113**

**Iteration 354: Best Cost = 113**

**Iteration 355: Best Cost = 113**

**Iteration 356: Best Cost = 113**

**Iteration 357: Best Cost = 113**

**Iteration 358: Best Cost = 113**

**Iteration 359: Best Cost = 113**

**Iteration 360: Best Cost = 113**

**Iteration 361: Best Cost = 113**

**Iteration 362: Best Cost = 113**

**Iteration 363: Best Cost = 113**

**Iteration 364: Best Cost = 113**

**Iteration 365: Best Cost = 113**

**Iteration 366: Best Cost = 113**

**Iteration 367: Best Cost = 113**

**Iteration 368: Best Cost = 113**

**Iteration 369: Best Cost = 113**

**Iteration 370: Best Cost = 113**

**Iteration 371: Best Cost = 113**

**Iteration 372: Best Cost = 113**

**Iteration 373: Best Cost = 113**

**Iteration 374: Best Cost = 113**

**Iteration 375: Best Cost = 113**

**Iteration 376: Best Cost = 113**

**Iteration 377: Best Cost = 113**

**Iteration 378: Best Cost = 113**

**Iteration 379: Best Cost = 113**

**Iteration 380: Best Cost = 113**

**Iteration 381: Best Cost = 113**

**Iteration 382: Best Cost = 113**

**Iteration 383: Best Cost = 113**

**Iteration 384: Best Cost = 113**

**Iteration 385: Best Cost = 113**

**Iteration 386: Best Cost = 113**

**Iteration 387: Best Cost = 113**

**Iteration 388: Best Cost = 113**

**Iteration 389: Best Cost = 113**

**Iteration 390: Best Cost = 113**

**Iteration 391: Best Cost = 113**

**Iteration 392: Best Cost = 113**

**Iteration 393: Best Cost = 113**

**Iteration 394: Best Cost = 113**

**Iteration 395: Best Cost = 113**

**Iteration 396: Best Cost = 113**

**Iteration 397: Best Cost = 113**

**Iteration 398: Best Cost = 113**

**Iteration 399: Best Cost = 113**

**Iteration 400: Best Cost = 113**

**Iteration 401: Best Cost = 113**

**Iteration 402: Best Cost = 113**

**Iteration 403: Best Cost = 113**

**Iteration 404: Best Cost = 113**

**Iteration 405: Best Cost = 113**

**Iteration 406: Best Cost = 113**

**Iteration 407: Best Cost = 113**

**Iteration 408: Best Cost = 113**

**Iteration 409: Best Cost = 113**

**Iteration 410: Best Cost = 113**

**Iteration 411: Best Cost = 113**

**Iteration 412: Best Cost = 113**

**Iteration 413: Best Cost = 113**

**Iteration 414: Best Cost = 113**

**Iteration 415: Best Cost = 113**

**Iteration 416: Best Cost = 113**

**Iteration 417: Best Cost = 113**

**Iteration 418: Best Cost = 113**

**Iteration 419: Best Cost = 113**

**Iteration 420: Best Cost = 113**

**Iteration 421: Best Cost = 113**

**Iteration 422: Best Cost = 113**

**Iteration 423: Best Cost = 113**

**Iteration 424: Best Cost = 113**

**Iteration 425: Best Cost = 113**

**Iteration 426: Best Cost = 113**

**Iteration 427: Best Cost = 113**

**Iteration 428: Best Cost = 113**

**Iteration 429: Best Cost = 113**

**Iteration 430: Best Cost = 113**

**Iteration 431: Best Cost = 113**

**Iteration 432: Best Cost = 113**

**Iteration 433: Best Cost = 113**

**Iteration 434: Best Cost = 113**

**Iteration 435: Best Cost = 113**

**Iteration 436: Best Cost = 113**

**Iteration 437: Best Cost = 113**

**Iteration 438: Best Cost = 113**

**Iteration 439: Best Cost = 113**

**Iteration 440: Best Cost = 113**

**Iteration 441: Best Cost = 113**

**Iteration 442: Best Cost = 113**

**Iteration 443: Best Cost = 113**

**Iteration 444: Best Cost = 113**

**Iteration 445: Best Cost = 113**

**Iteration 446: Best Cost = 113**

**Iteration 447: Best Cost = 113**

**Iteration 448: Best Cost = 113**

**Iteration 449: Best Cost = 113**

**Iteration 450: Best Cost = 113**

**Iteration 451: Best Cost = 113**

**Iteration 452: Best Cost = 113**

**Iteration 453: Best Cost = 113**

**Iteration 454: Best Cost = 113**

**Iteration 455: Best Cost = 113**

**Iteration 456: Best Cost = 113**

**Iteration 457: Best Cost = 113**

**Iteration 458: Best Cost = 113**

**Iteration 459: Best Cost = 113**

**Iteration 460: Best Cost = 113**

**Iteration 461: Best Cost = 113**

**Iteration 462: Best Cost = 113**

**Iteration 463: Best Cost = 113**

**Iteration 464: Best Cost = 113**

**Iteration 465: Best Cost = 113**

**Iteration 466: Best Cost = 113**

**Iteration 467: Best Cost = 113**

**Iteration 468: Best Cost = 113**

**Iteration 469: Best Cost = 113**

**Iteration 470: Best Cost = 113**

**Iteration 471: Best Cost = 113**

**Iteration 472: Best Cost = 113**

**Iteration 473: Best Cost = 113**

**Iteration 474: Best Cost = 113**

**Iteration 475: Best Cost = 113**

**Iteration 476: Best Cost = 113**

**Iteration 477: Best Cost = 113**

**Iteration 478: Best Cost = 113**

**Iteration 479: Best Cost = 113**

**Iteration 480: Best Cost = 113**

**Iteration 481: Best Cost = 113**

**Iteration 482: Best Cost = 113**

**Iteration 483: Best Cost = 113**

**Iteration 484: Best Cost = 113**

**Iteration 485: Best Cost = 113**

**Iteration 486: Best Cost = 113**

**Iteration 487: Best Cost = 113**

**Iteration 488: Best Cost = 113**

**Iteration 489: Best Cost = 113**

**Iteration 490: Best Cost = 113**

**Iteration 491: Best Cost = 113**

**Iteration 492: Best Cost = 113**

**Iteration 493: Best Cost = 113**

**Iteration 494: Best Cost = 113**

**Iteration 495: Best Cost = 113**

**Iteration 496: Best Cost = 113**

**Iteration 497: Best Cost = 113**

**Iteration 498: Best Cost = 113**

**Iteration 499: Best Cost = 113**

**Iteration 500: Best Cost = 113**

**>> BestSol.Sol**

**ans =**

**L: {4x1 cell}**

**ST: [20x1 double]**

**PT: [20x1 double]**

**FT: [20x1 double]**

**MCT: [4x1 double]**

**Cmax: 113**

**>> BestSol.Position**

**21,22,23 جدا کننده هستند بین ماشین ها**

**ans =**

**Columns 1 through 18**

**9 17 18 2 8 ماشین اول21 10 11 6 12 ماشین دوم23 1 4 19 15 5 ماشین سوم22 7**

**Columns 19 through 23**

**20 3 14 16 13**

**ماشین چهارم**

**>>**