Μέθοδοι Βελτιστοποίησης

Homework 1 EA

Λογοθέτης Μιχαήλ mc18687

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΙΟ

Σχολή Μηχανολόγων Μηχανικών

Λογοθέτης Μιχαήλ

02118687

30 Νοεμβρίου 2021

Branching Pipes Problem

Εισαγωγή

Για να απαντήσουμε στα ερωτήματα της άσκησης χρησιμοιήθηκε το λογισμικό EASY.Σκοπός του προβλήματος είναι η μελέτη της στοχαστικής μεθόδου των εξελικτικών αλγορίρθων EA. Αρχικά με βάση τα στοιχεία της εκφώνησης δημιούργησα ένα κώδικα ο οποίος με την επαναληπτική μέθοδο Hardy-Cross υπολογίζει την παροχή (branchP.f95 , γραμμένο σε FORTRAN90 και δίνεται στο παράρτημα). Στην συνέχεια κάνουμε compile τον κώδικα για να πάρουμε το εκτελέσιμο αρχείο branchP.exe το οποίο είναι αυτό που θα περάσουμε στο λογισμικό EASY και θα μας δώσει τα αποτελέσματα του προβλήματος(task.res).

Στοιχεία εκφώνησης που αλλάζουν ανάλογα τον φοιτητή:

$d_{δε}$=0.055 m

$d_{ακ}$=0.144 m

$d_{κλ}$=0.108 m

Άρα οι υπόλοιπες διάμετροι $d_{κλ}$,$d_{λγ}$,$d_{λμ}$ θα βρίσκονται μεταξύ 0.055-0.144 [m]

Single Objective Optimization

Ρύθμιση παραμέτρων του EASY, θέτοντας τις μεταβλητές όπως φαίνεται παρακάτω με σκοπό οι διακριτές λύσεις να απέχουν μεταξύ τους λογική/εφικτή απόσταση.Η επιλογή των 10 bits έγινε έτσι ώστε να έχουμε 2^{10}=1024 εφικτές λύσεις για την κατασκευή των σωληνώσεων.

ID	Min	Max	Const	bits	Comment
1	0.055	0.144	☐	10	Dkl
2	0.055	0.144	☐	10	Dlg
3	0.055	0.144	☐	10	Dlm
4	75.0	150.0	☐	10	xk
5	-50.0	50.0	☐	10	yk
6	0.0	50.0	☐	10	zk
7	125.0	150.0	☐	10	xl
8	-30.0	30.0	☐	10	yl
9	-50.0	50.0	☐	10	zl
10	50.0	125.0	☐	10	xm

Το πρώτο πρόβλημα βελτιστοποίησης είναι η μέγιστη απορροή απο την δεξαμενή Α και να τροφοδοτεί με ίσες παροχές τις άλλες 3 δεξαμενές.Πρόκειται για πρόβλημα μεγιστοποίησης και ο EASY λύνει μόνο προβλήματα ελαχιστοποίησης. Επομένως με τις κατάλληλες προσαρμογές μετατρέπουμε το πρόβλημα σε ελαχιστοποίηση του $-Q_a$.

Οι περιορισμοί γράφονται στον κώδικα και αργότερα μεταφέρονται στον EASY με την εξής μορφή.

α) Q_a , Q_b , Q_c , $Q_d > 0$

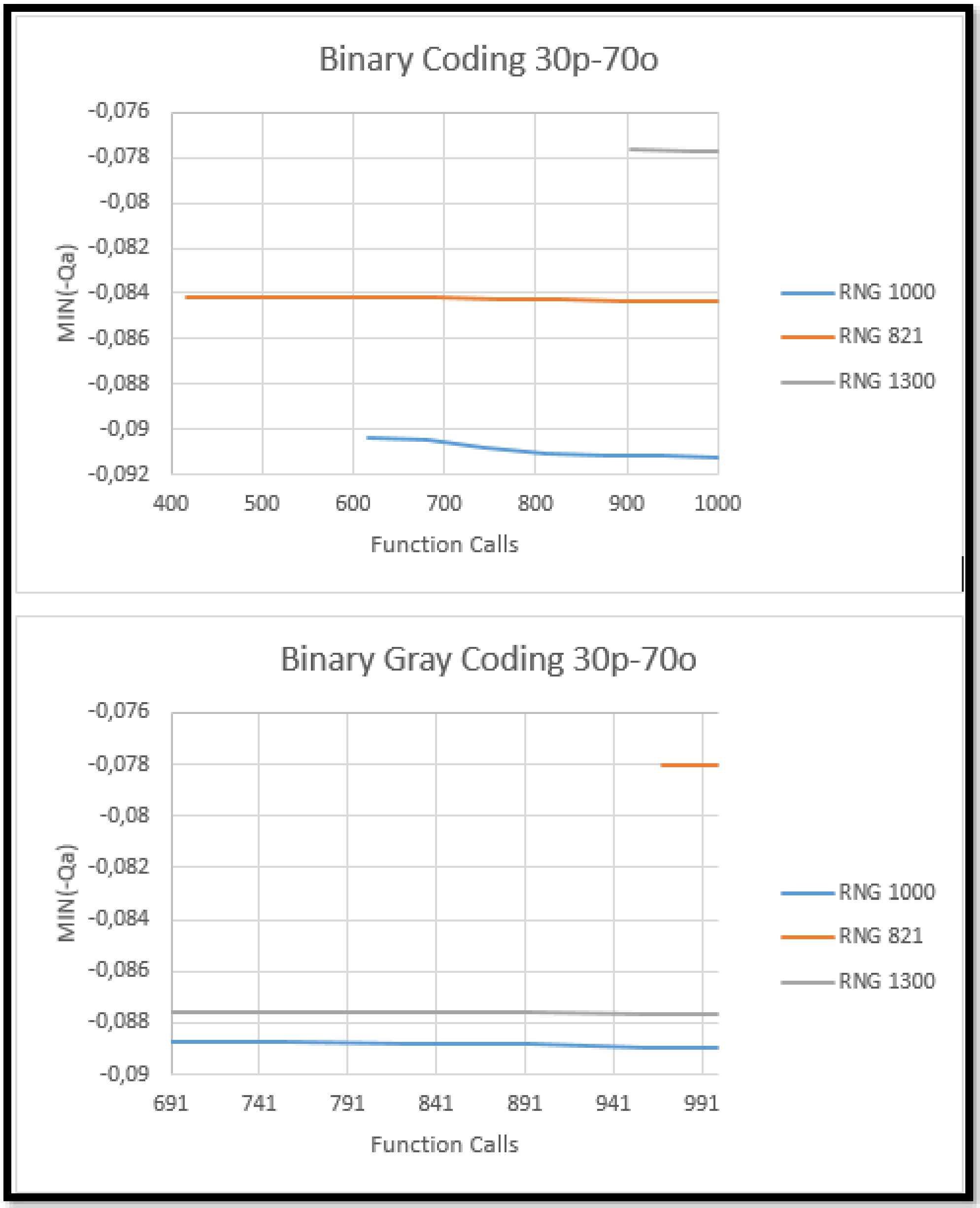
β) $|Q_C-Q_b|=|Q_c-Q_d| \cong 10^{-4}$

ID	Nominal threshold	Relaxed threshold	Amplification factor	Comment
1	0.0	0.1	3.0	-Qa
2	0.0	0.03	3.0	-Qb
3	0.0	0.03	3.0	-Qc
4	0.0	0.03	3.0	-Qd
5	1.0E-4	0.0010	3.0	Qc-Qb
6	1.0E-4	0.0010	3.0	Qc-Qd

Έγιναν $3 \times 3 \times 2$ τρεξίματα στο EASY (3 διαφορετικές ψευδογεννήτριες τυχαίων αριθμών PRNG , 3 διαφορετικοί συνδιασμοί γονέων(parents-p)-παιδιών(offspring-o) και 2 διαφορετικοί μέθοδοι κωδικοποίησης Binary-Binary Gray).Τα παρακάτω διαγράμματα προέκυψαν μέσω EXCEL απο το αρχείο EA_L1_GX.log ,όπου Χ συμβολίζει τον αριθμό της γενιάς . Για το Crossover χρησιμοποίησα σταθερά πιθανότητα 0.9-Two Point Mode και 0.9-Two Point/Var Mode , ενώ για το Mutation ένα εύρος πιθανότητας 0.02-0.06.

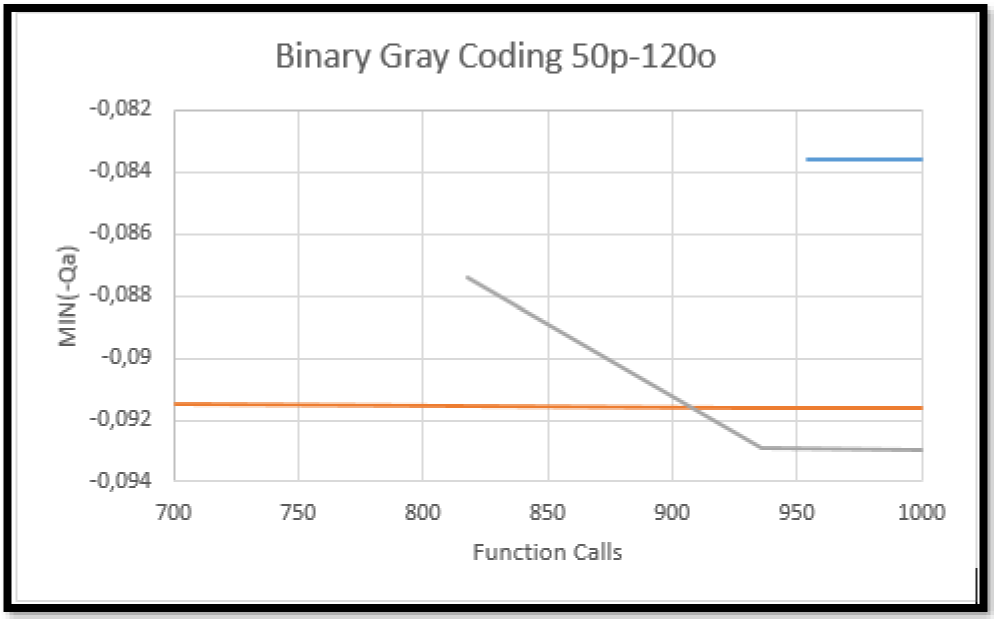
<u>ΣΧΗΜΑ 1</u>

ΣΧΗΜΑ 2



Παρατήρηση : Στην προσπάθεια μου να βρώ βέλτιστη λύση πάντα οι πρώτες κλήσεις της αντικειμενικής συνάρτησης ήταν εντός περιορισμών με αποτέλεσμα ο EASY να δίνει την τιμή $1.7*10^{305}$(penalized),γι’ αυτο το λόγο έχω αφαιρέσει στα διαγράμματα ένα εύρος function calls.Επιπλέον παρακάτω δεν παρατίθεται το διάγραμμα για Binary Coding 50p-120o γιατί είτε η βέλτιστη τιμή ήταν μικρή($\approx -0{,}07$) σε σχέση με τις υπόλοιπες είτε δεν προέκυπτε κάποια

ΣΧΗΜΑ 3



Τελικά η καλύτερη και ταχύτερη λύση σε σύγκριση με τις υπόλοιπες είναι η PRNG=1000,Binary Coding και 15p-30o , όπως φαίνεται και στο Σχήμα 1.

Variables= [D_{kb} ,D_{lg} ,D_{lm} ,x_k ,y_k ,z_k ,x_l ,y_l ,z_l ,x_m] [m]

Variables(best)= [0.06483089 , 0.063438905 , 0.130080156 , 127.05278592375 , -0.73313782991202 , 29.716520039101 , 138.14760508309 , 3.782991202346 , 30.254154447703 , 61.436950146628]

$-Q_a$(min)= -0,094988951 [m^3/s]

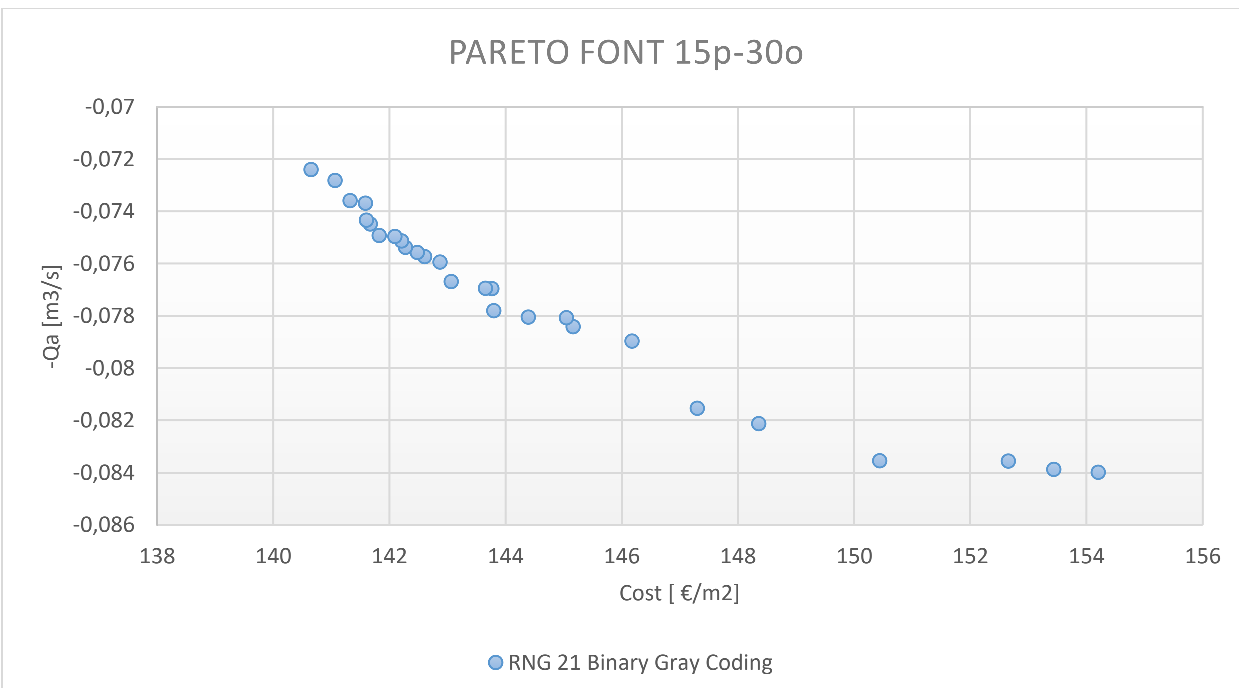
Παρατήρηση: Η βελτιστοποίηση Binary/Binary Gray 15p-30o κυριάρχησε μεταξύ των άλλων μεθόδων στην ελαχιστοποίηση της παροχής με λιγότερες κλήσεις της αντικειμενικής συνάρτησης. Η αύξηση του μεγέθους του πληθυσμού για το συγκεκριμένο πρόβλημα φαίνεται να επηρεάζει αρνητικά τον χρόνο σύγκλισης (εφόσον θεωρηθεί πως ο χρόνος μετριέται σε Function Calls , ότι δηλαδή τα 1000 Function Calls κάνουν τον ίδιο χρόνο σε κάθε περίπτωση). Παρόλα αυτά η αύξηση του μεγέθους πληθυσμού επιφέρει προφανή μείωση στις γενιές που μεσολαβούν μέχρι την λύση, κάτι το οποίο είναι λογικό καθώς γίνονται λιγότερες κλήσεις της συνάρτησης ανά γενιά.

Multi Objective Optimization

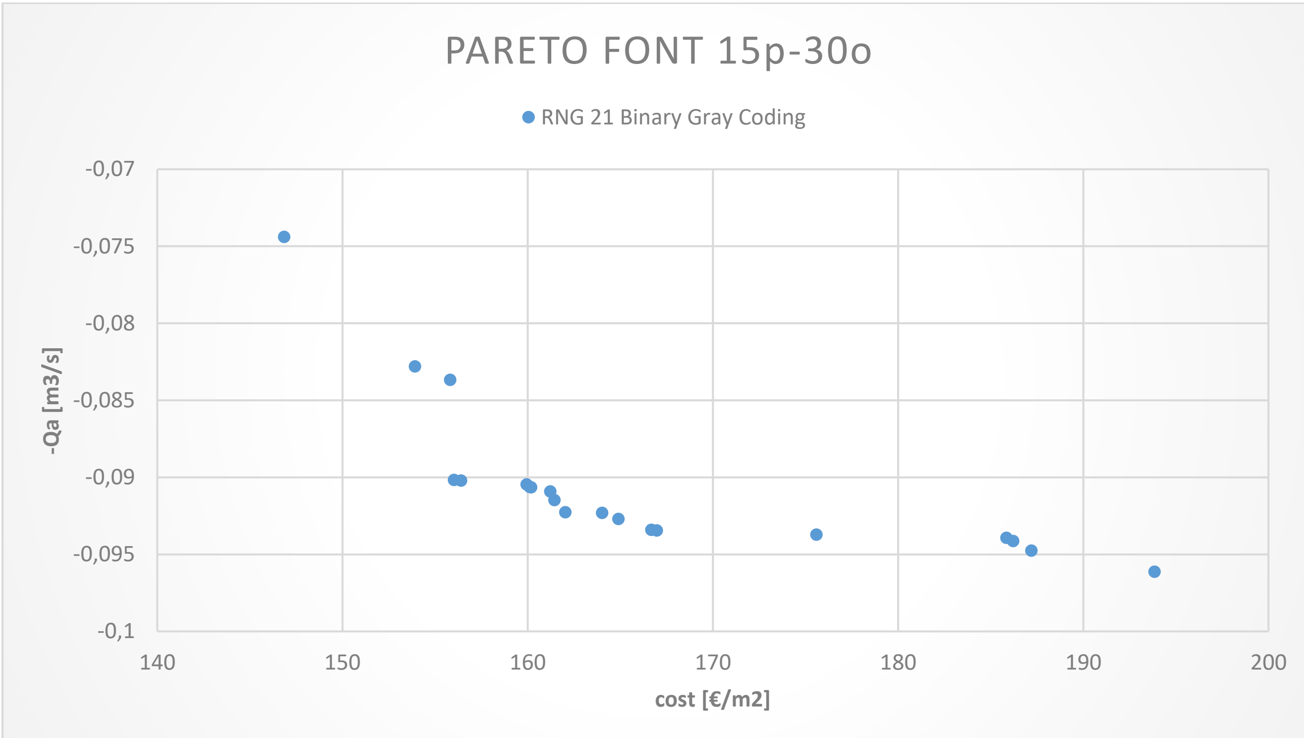
Για το πρόβλημα πολλαπλών στόχων που στην περίπτωση μας είναι 2 στόχων καλούμαστε να ελαχιστοποιήσουμε το κόστος των σωλήνων υποθέτοντας οτι το πάχος των σωλήνων είναι μηδενικό.Τροποποιούμε το branch.f95 κατάλληλα και ρυθμίζουμε τον EASY για να ανταποκρίνεται στο πρόβλημα.Αυξάνουμε τον αριθμό των elite και τρέχουμε το λογισμικό για να μας δώσει το PARETO FONT. Συγκεκριμένα μετά από κάποια τρεξίματα εισάγω μέσα στην αρχικοποίηση του προβλήματος μία τιμή που έχει βρεθεί από προηγούμενη αναζήτηση για την οποία ξέρω πως τηρούνται οι περιορισμοί προκειμένου ο EA να ξέρει που θα αναζητήσει elites και να μην σκοτώνει τις περισσότερες αρχικοποιήσεις που δίνει στο πρόβλημα. Το αρχείο το δίνετε παρακάτω:


EA_Lini - Notepad
File Edit Format View Help
1
0.06483089 0.063438905 0.130080156 127.05278592375 -0.73313782991202 29.716520039101 138.14760508309 3.782991202346 30.254154447703 61.436950146628

ΣΧΗΜΑ 4



<u>ΣΧΗΜΑ 5</u>



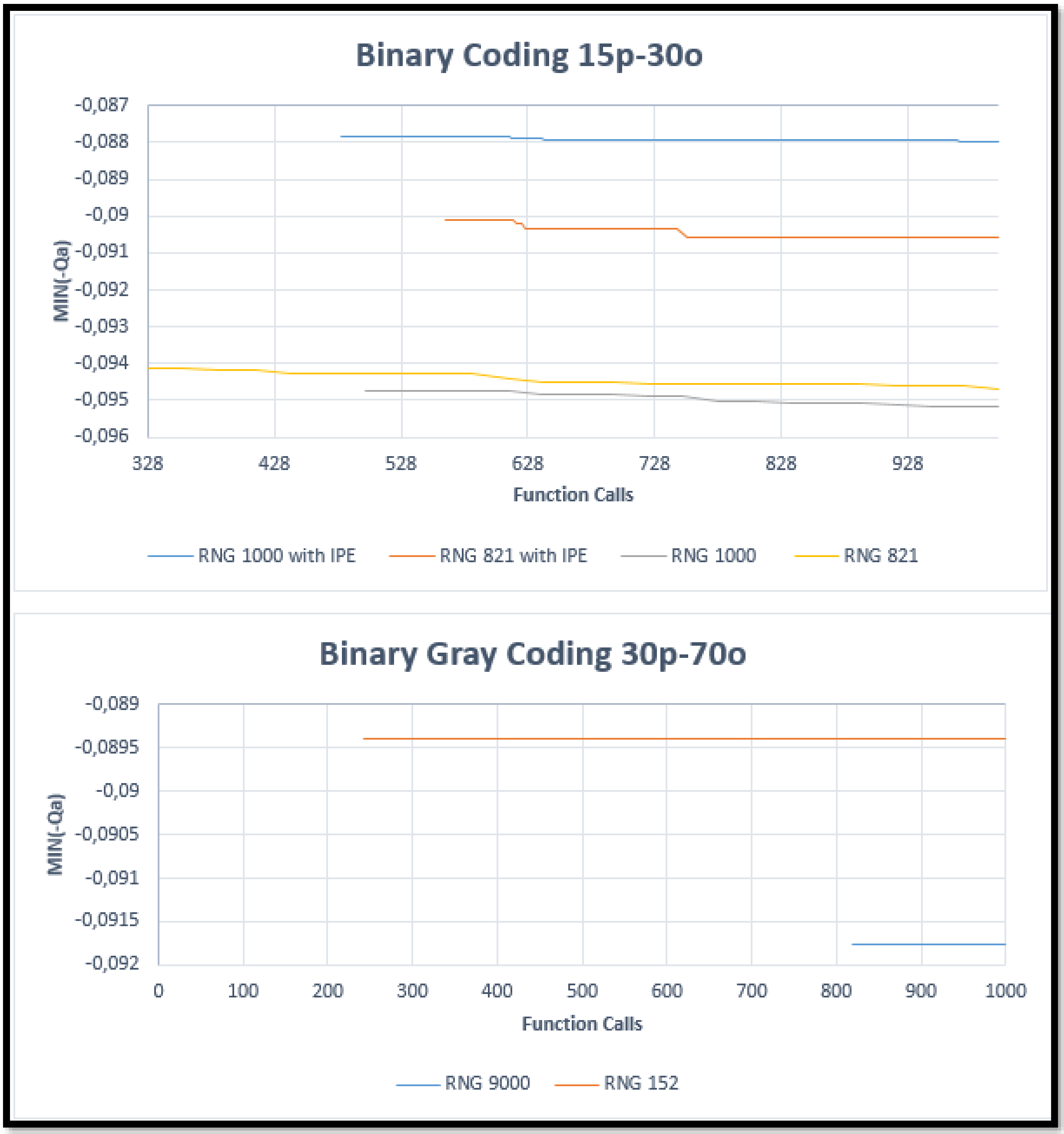
Παρατήρηση: Αύξησα τα Function Calls στα 3000

<u>SOO with IPE</u>

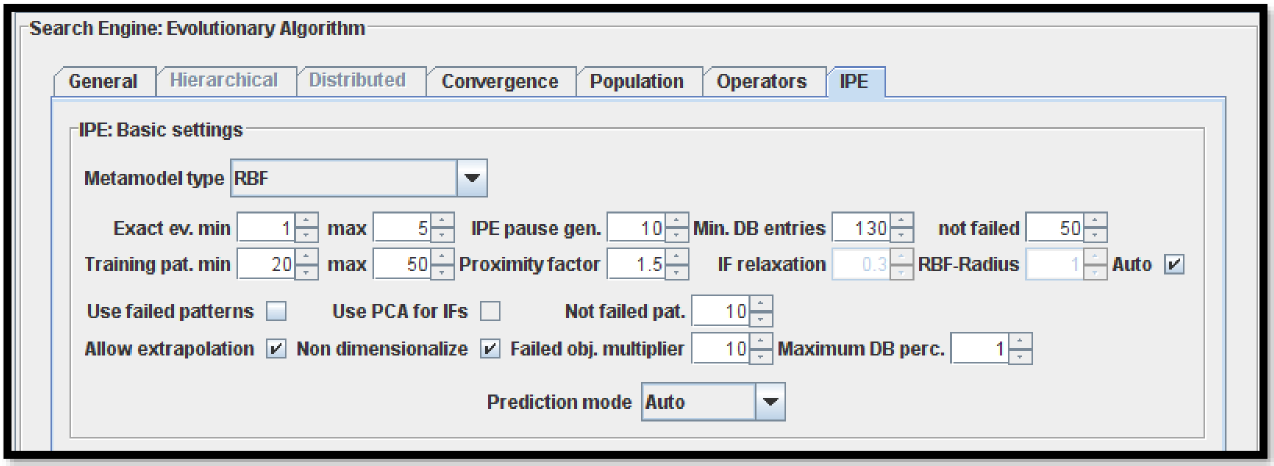
Low Cost Pre-Evaluation

Ο τύπος Metamodel που χρησιμοποιήθηκε είναι RBF (Radial Basis Function)

<u>ΣΧΗΜΑ 6</u>



Οι ρυθμίσεις που χρησιμοποίησα για κάθε τρέξιμο

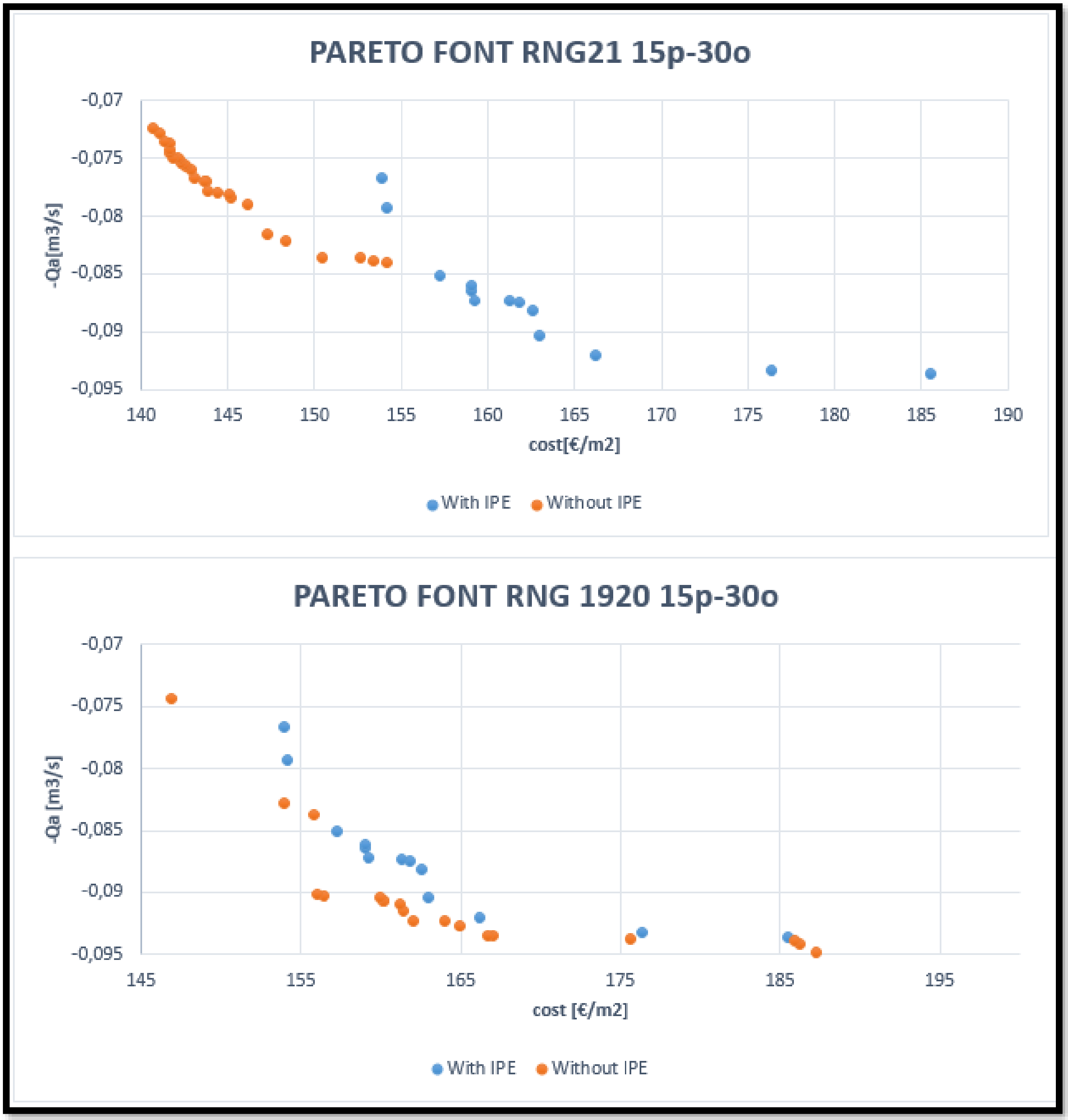


Παρατήρηση : Το τρέξιμο με IPE έχουμε κάποιες λύσεις οι οποίες δεν μειώνουν την $-Q_a$. Αυτό είναι χαρακτηριστικό για τον τρόπο που δουλεύει ο IPE καθώς το Pre-Evaluation προβλέπει πως η λύση που θα προκύψει είναι χειρότερη επομένως δεν την λαμβάνει υπόψιν, με αποτέλεσμα να μην μπαίνει στο Evaluation. Παρόλα αυτά το τρέξιμο με IPE μειώνει σημαντικά τα Evaluations που χρειάζονται για να συγκλίνει ο αλγόριθμος στην βέλτιστη λύση. Στο συγκεκριμένο πρόβλημά μπορεί να μην είναι πολύ η μείωση αλλά σε κάποιο μελλοντικό project η μείωση ,έστω και σε αυτό το ποσοστό, του υπολογιστικού κόστους είναι καθοριστική.

MOO with IPE

Ξανατρέχουμε το πρόβλημα 2 στόχων με τους ίδους περιορισμούς και το IPE.

ΣΧΗΜΑ 7



ΠΑΡΑΡΤΗΜΑ ΚΩΔΙΚΑ

```
program branchP
implicit double precision (a-h, o-z)
!
open(1,file="task.dat")
     read(1,*) Ndv
     read(1,*) dkb
     read(1,*) dlg
     read(1,*) dlm
     read(1,*) xk
     read(1,*) yk
     read(1,*) zk
     read(1,*) xl
     read(1,*) yl
     read(1,*) zl
     read(1,*) xm
close(1)
pi=3.14159265359
g=9.81
fe=0.008
f=0.01
dde=0.055
dak=0.144
dkl=0.108
zm=(17./75.)*xm-111.333
Sak=dsqrt((xk-50.)**2+(yk)**2+(zk-50.)**2)
Skb=dsqrt((xk-175.)**2+(yk)**2+(zk)**2)
Skl=dsqrt((xk-xl)**2+(yk-yl)**2+(zk-zl)**2)
Slg=dsqrt((xl-175)**2+(yl+100)**2+(zl+100)**2)
Slm=dsqrt((xl-xm)**2+(yl-100)**2+(zl-zm)**2)
Sdm=dsqrt((xm-50.)**2+(ym-100.)**2+(zm+100.)**2)
Sme=dsqrt((xm-125.)**2+(ym-100.)**2+(zm+83.)**2)
cost=pi*(Sak*dak+Skb*dkb+Skl*dkl+Slg*dlg+Slm*dlm+(Sdm+Sme)*dde)
open(1,file="pipes")
     write(1,*) "Lak (m) =",Sak
     write(1,*) "Lbk (m) =",Skb
     write(1,*) "Lkl (m) =",Skl
     write(1,*) "Llg (m) =",Slg
     write(1,*) "Llm (m) =",Slm
     write(1,*) "Ldm (m) =",Sdm
     write(1,*) "Lme (m) =",Sme
close(1)
!
aKak=(f*Sak*8.)/(pi**2*g*dak**5)          !f*Sak/dak**5*8/pi**2*g
aKkl=(f*Skl*8.)/(pi**2*g*dkl**5)
aKkb=(f*Skb*8.)/(pi**2*g*dkb**5)
aKlg=(f*Slg*8.)/(pi**2*g*dlg**5)
Qb = Qa/3.
Qc = Qb
Qd = Qb
Qk = 2.*Qa/3.
!write(*,*) "Starting Hardy-Cross Method"
!write(*,*)
open(1,file="res")
do i=1,100
     erMax = -1.
     if (i.ne.1) then
          Qa = Qan
          Qb = Qbn
          Qc = Qcn
          Qd = Qdn
          Qk = Qkn
     endif
     write(*,*) "  Iteration",i
     a=-aKak*abs(Qa)*Qa-aKkb*abs(Qb)*Qb+50.
     b=aKak*abs(Qa)+aKkb*abs(Qb)
     dQ1 = -0.5*(a/b)
     a=-aKlg*abs(Qc)*Qc-aKkl*abs(Qk)*Qk+aKkb*abs(Qb)*Qb+100.
     b=aKlg*abs(Qc)+aKkl*abs(Qk)+aKkb*abs(Qb)
     dQ2= -0.5*(a/b)
     a= aKak*abs(Qa)*Qa+aKkl*abs(Qk)*Qk+(aKlm+aKdm)*abs(Qd)*Qd-150.
     b= aKak*abs(Qa)+aKkl*abs(Qk)+(aKlm+aKdm)*abs(Qd)
     dQ3=-0.5*(a/b)
     write(1,*) i, dQ1, dQ2, dQ3
     !write(*,*) "    dQ2 =",dQ2
    ! write(*,*) "    dQ3 =",dQ3
     Qan = Qa-dQ1+dQ3
     Qbn = Qb-dQ1+dQ2
     Qcn = Qc-dQ2
     Qdn = Qd+dQ3
     Qkn = Qk+dQ3-dQ2
     if (abs(dQ1)>erMax) erMax = abs(dQ1)
     if (abs(dQ2)>erMax) erMax = abs(dQ2)
     if (abs(dQ3)>erMax) erMax = abs(dQ3)
     if (erMax<1.e-10) exit
enddo
close(1)
!write(*,*) "Ending Hardy-Cross Method"
!write(*,*)
write(*,*)"Fluxes:"
write(*,*) "   Qa =",Qan, " m^3/s"
write(*,*) "   Qb =",Qbn, " m^3/s"
write(*,*) "   Qc =",Qcn, " m^3/s"
write(*,*) "   Qd =",Qdn, " m^3/s"
write(*,*) "   Qk =",Qkn, " m^3/s"
!
open(1,file="task.res")
     write(1,*) -Qan
     write(1,*) cost
close(1)
!
open(1,file="task.cns")
     write(1,*) -Qan
     write(1,*) -Qbn
     write(1,*) -Qcn
     write(1,*) -Qdn
     write(1,*) abs(Qcn-Qbn)
     write(1,*) abs(Qcn-Qdn)
close(1)
!
end
```