Սեմինար

Ինտերնետ սոկետների կիրառումը զուգահեռ հաշվարկների համար NICA կոլայդերի Թվիս պարամետրերի համաձայնեցման խնդրի օրինակով

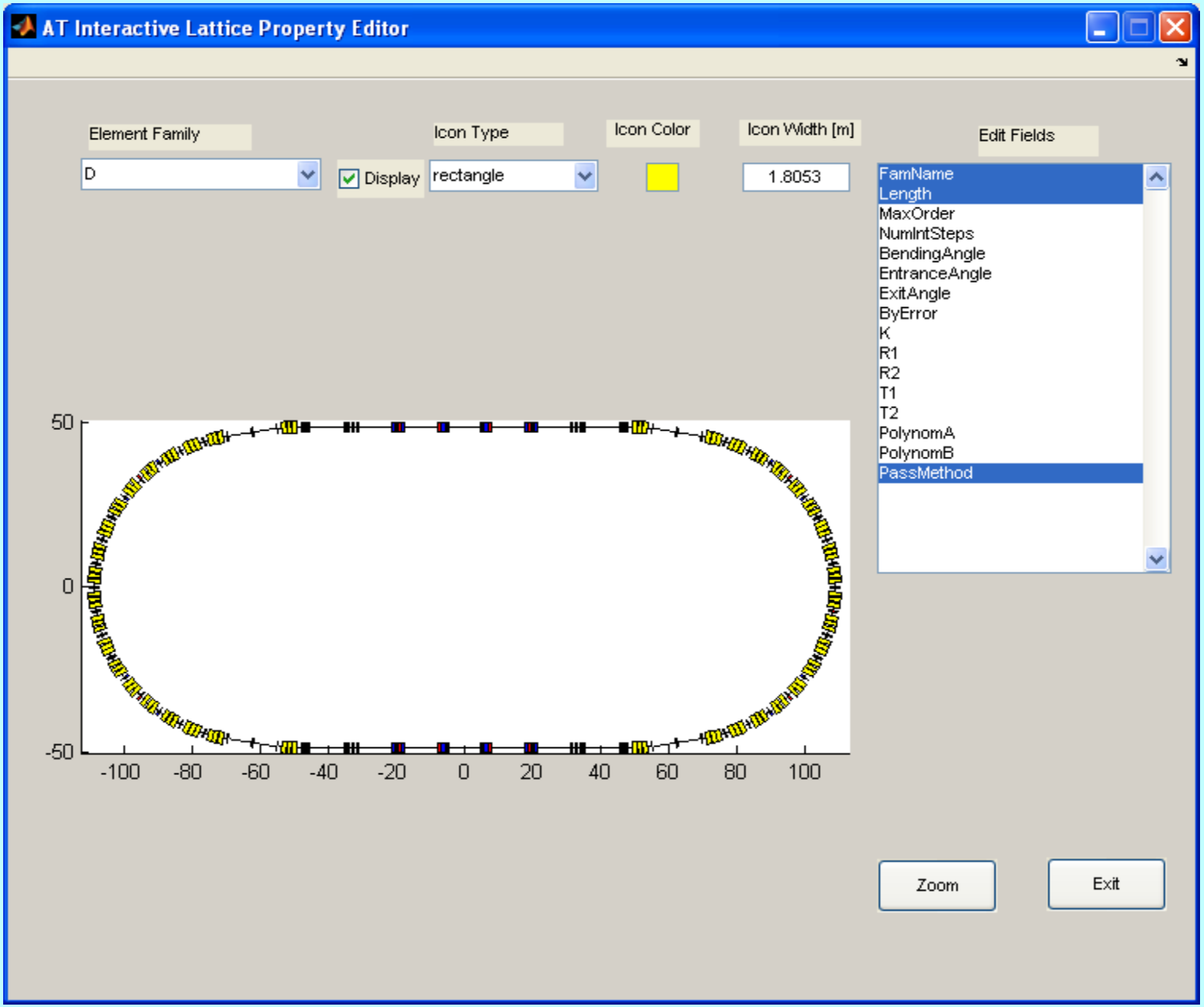
10 օգոստոս 2011թ.

Դ. Քալանթարյան (ԱԱԳԼ) և Ա. Այրիյան (ЛИТ ОИЯИ)

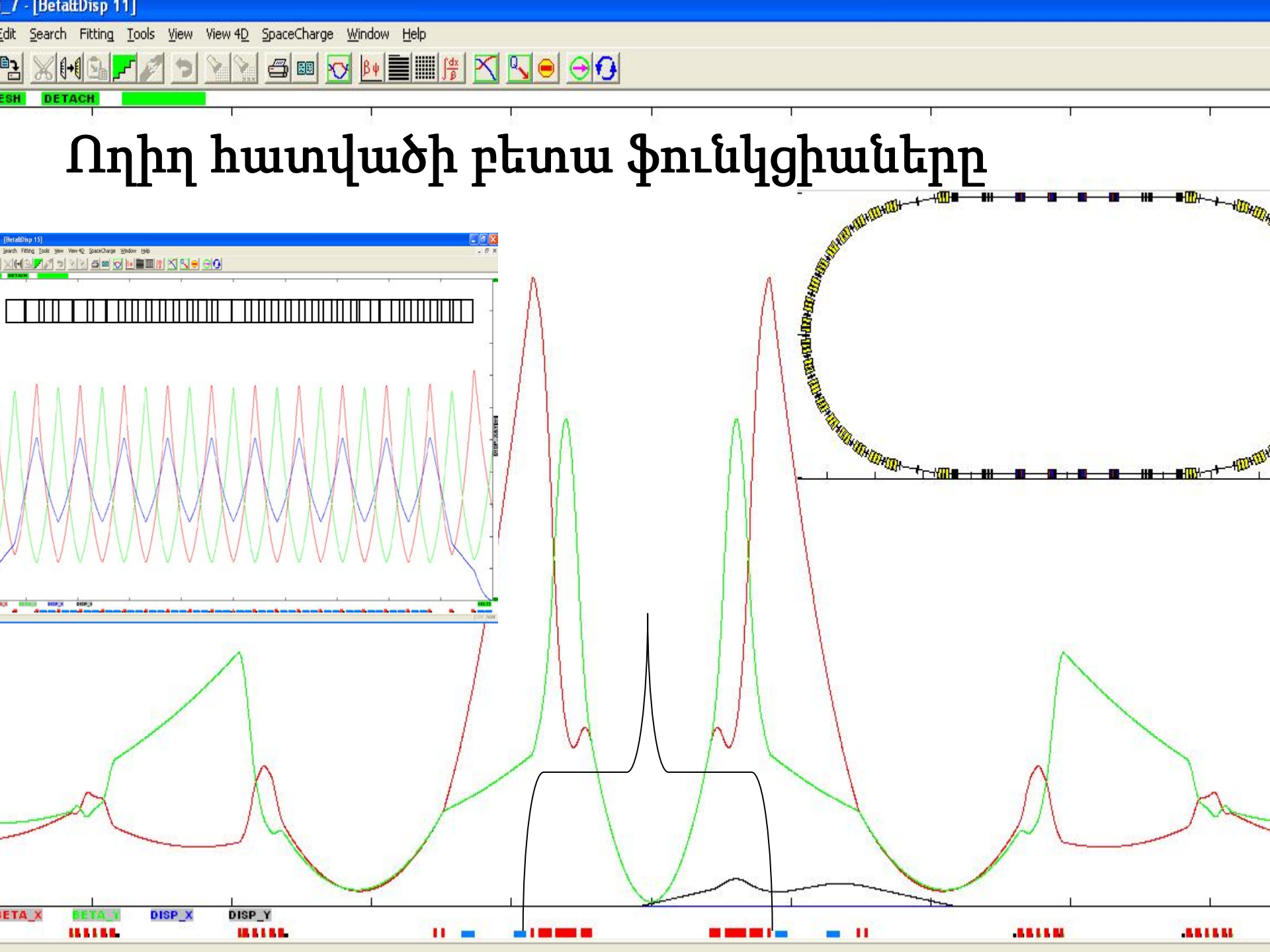
Բովանդակություն

1. Համաձայնեցման խնդիրը
2. Ինչ է սոկետը
3. Սոկետների օգտագործումը ստեղծելու համար հաշվարկման ցանց ծավալով հաշվարկներ կատարելու համար

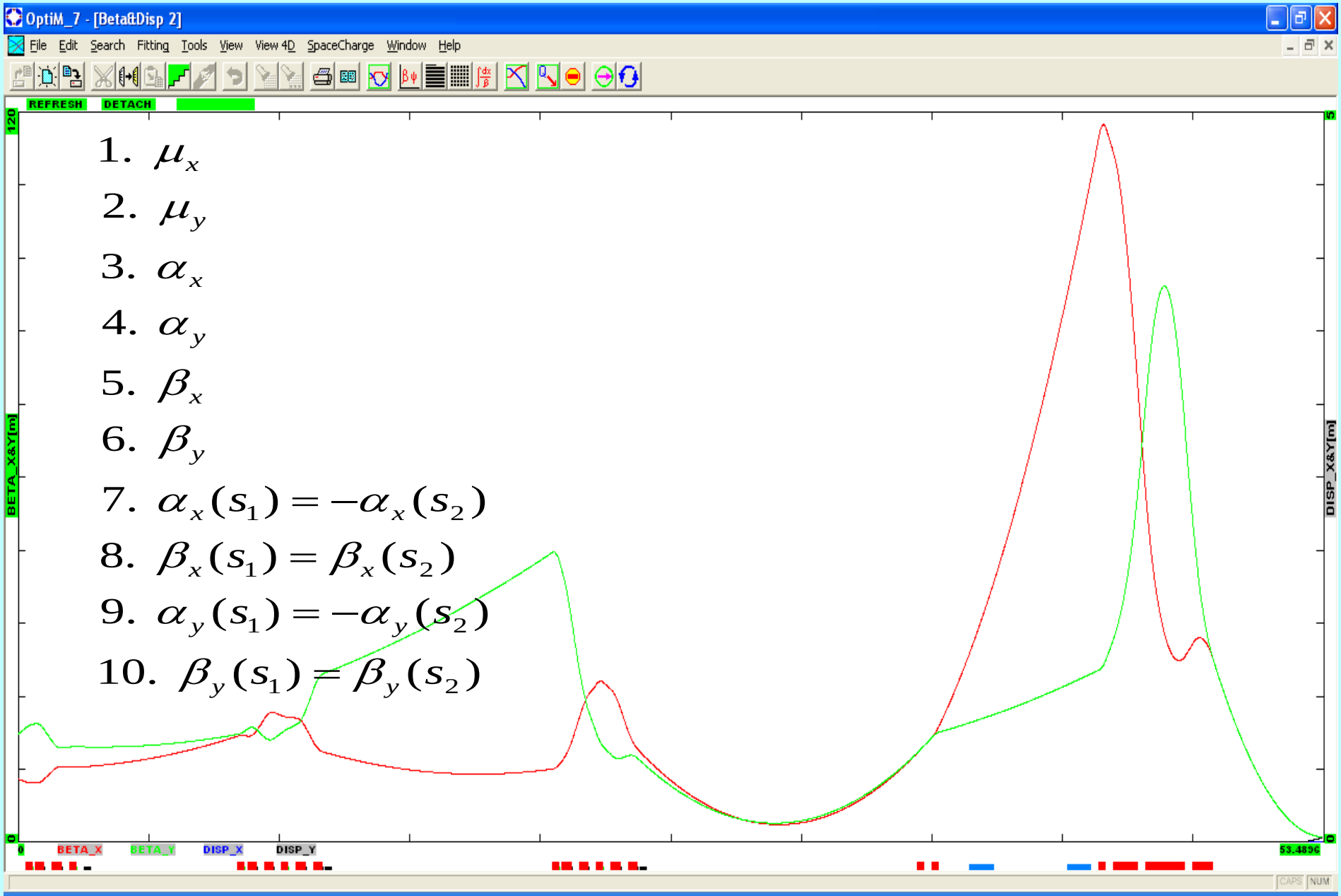
Պարամետրերի համաձայնեցման խնդիր



Ողող հատվածի բետա ֆունկցիաները

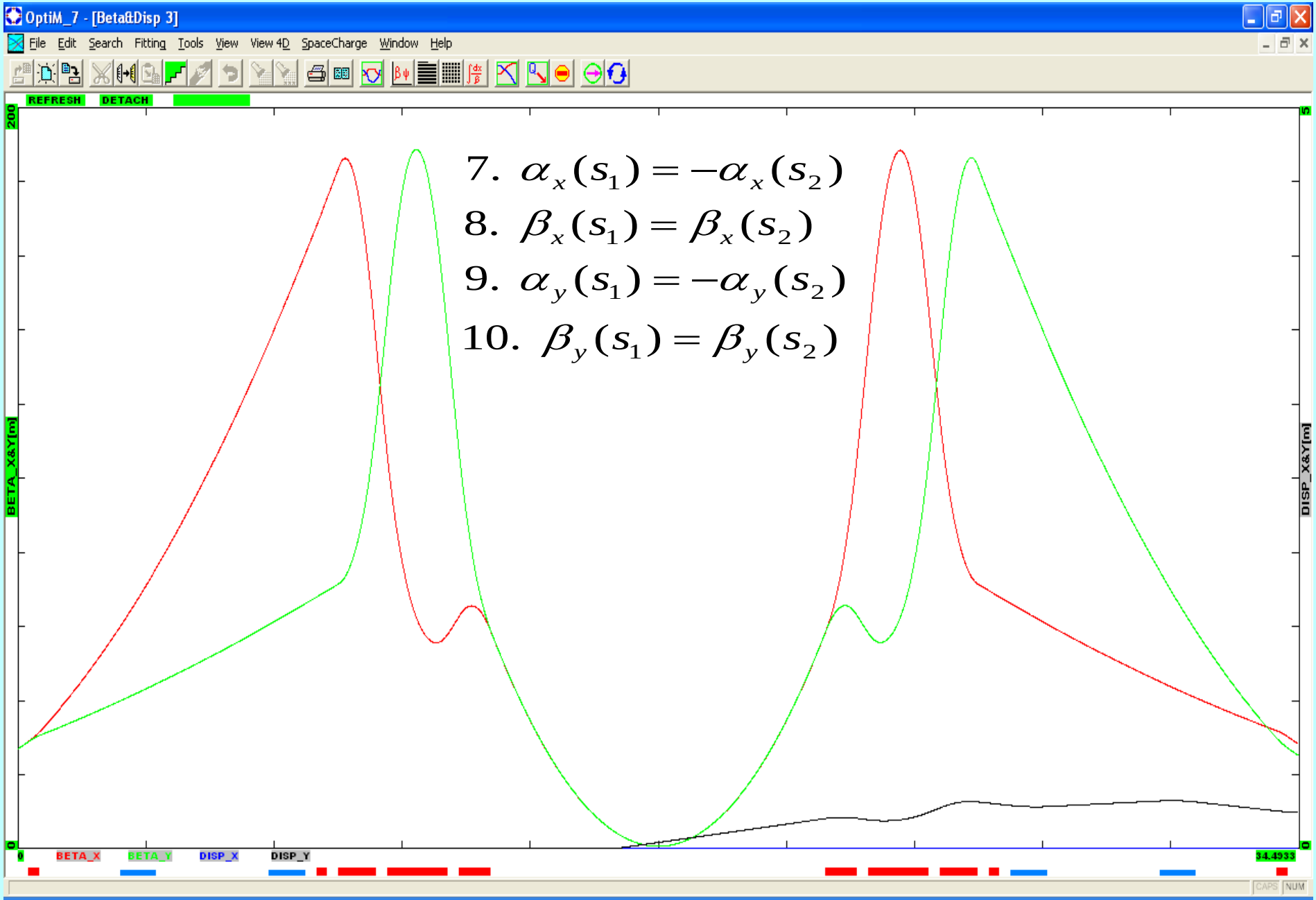


Պարամետրերը, որոնք պետք է համաձայնեցնել



1. μ_x
2. μ_y
3. α_x
4. α_y
5. β_x
6. β_y
7. $\alpha_x(s_1) = -\alpha_x(s_2)$
8. $\beta_x(s_1) = \beta_x(s_2)$
9. $\alpha_y(s_1) = -\alpha_y(s_2)$
10. $\beta_y(s_1) = \beta_y(s_2)$

Последние четыре параметры



7. $\alpha_x(s_1) = -\alpha_x(s_2)$

8. $\beta_x(s_1) = \beta_x(s_2)$

9. $\alpha_y(s_1) = -\alpha_y(s_2)$

10. $\beta_y(s_1) = \beta_y(s_2)$

խնդիրը լուծելու համար անհրաժեշտ իտերացիաների քանակը

$$P_1 = F_1(g_1,\ldots,g_{N_m}) = P_{10}$$

Տված համակարգը կունենա գոնե 1 լուծում, եթե Յակոբիի մատրիցի ռանգը հավասար լինի N_p

$$\ldots \quad \ldots \quad \ldots \quad \ldots$$

$$P_{N_p} = F_{N_p}(g_1,\ldots,g_{N_m}) = P_{N_p 0}$$

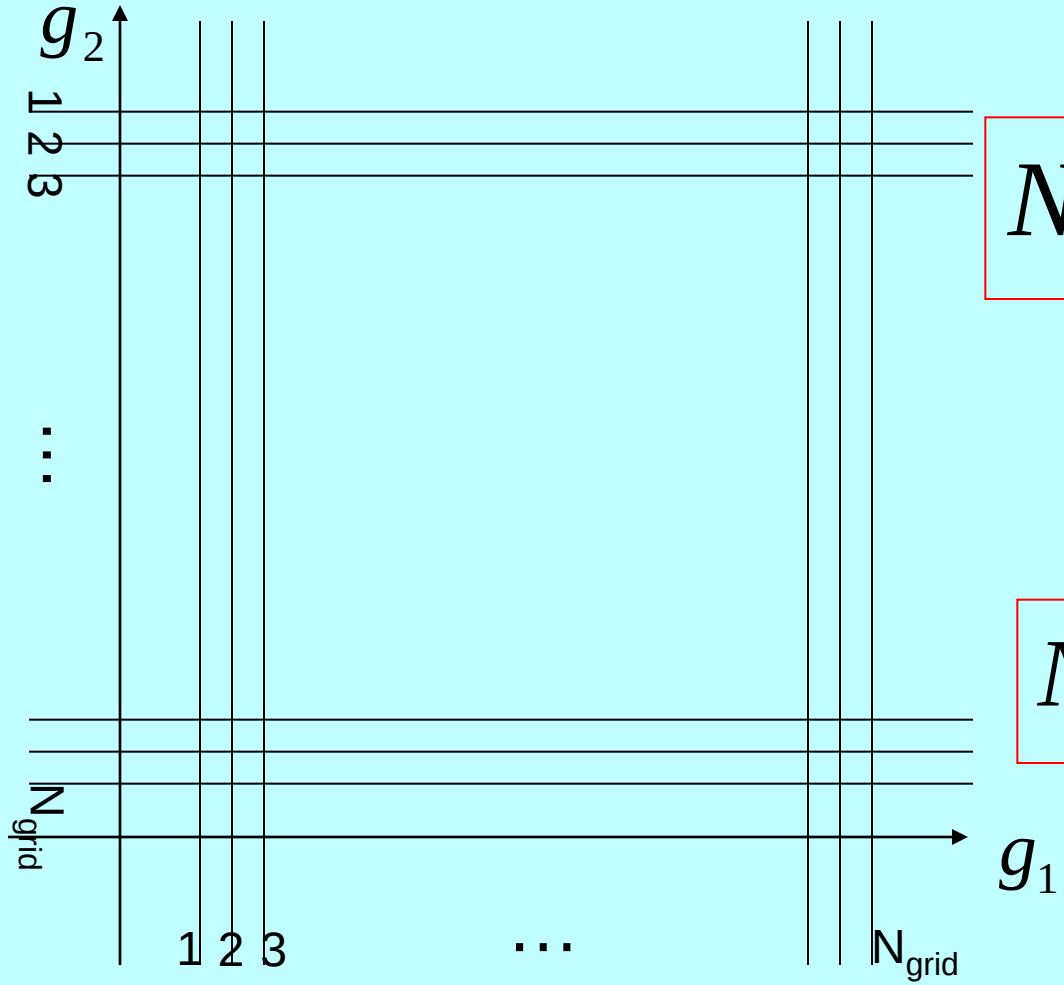
$$\boxed{rang(J) = rang\left(\frac{\partial P_i}{\partial g_j}\right) = N_p}$$

Սա նշանակում է, որ լուծում կունենանք միայն այն դեպքում, երբ $\mathbf{N_m >= N_p}$

Պարզ մոտեցման դեպքում, երբ կառուցվում է ցանց բոլոր գրադիենտներով և լուծումը փնտրվում է այդ ցանցի գագաթներում, ալգորիթմը կկատարի հետևյալ քանակով իտերացիաներ:

$$\boxed{N_{iter} = N_{grid}^{N_m}}$$

Օրինակ N_p=2 դեպքը



$$N_m = 2$$

$$N_{grid} = 1000$$

$$\boxed{N_{iter} = N_{grid}^{N_m} = 1e6}$$

$$N_m = 10$$

$$\boxed{N_{iter} = N_{grid}^{N_m} = 1e30}$$

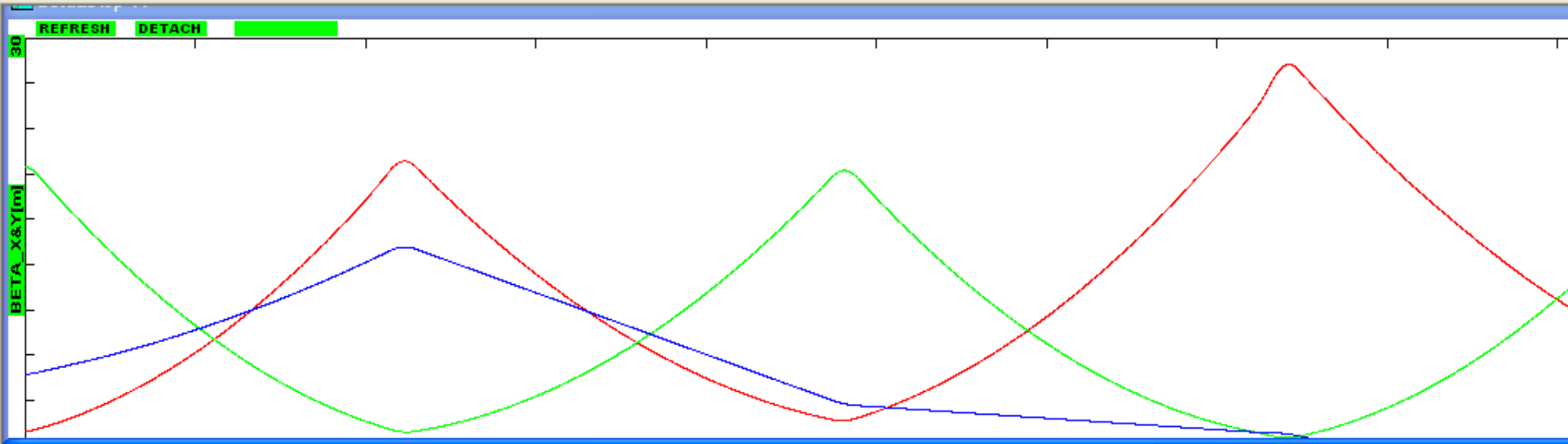
Պարամետրերի աղյուսակը, որոնք պետք է համաձայնեցնել և այն պարամետրերը, որոնք կարելի է այդ ցուցակից հանել

1. μ_x
2. μ_y
3. $\alpha_x(0)$
4. $\alpha_y(0)$
5. $\beta_x(\sim 35cm)$
6. $\beta_y(\sim 35cm)$
7. $\alpha_x(s_1) = -\alpha_x(s_2)$
8. $\beta_x(s_1) = \beta_x(s_2)$
9. $\alpha_y(s_1) = -\alpha_y(s_2)$
10. $\beta_y(s_1) = \beta_y(s_2)$

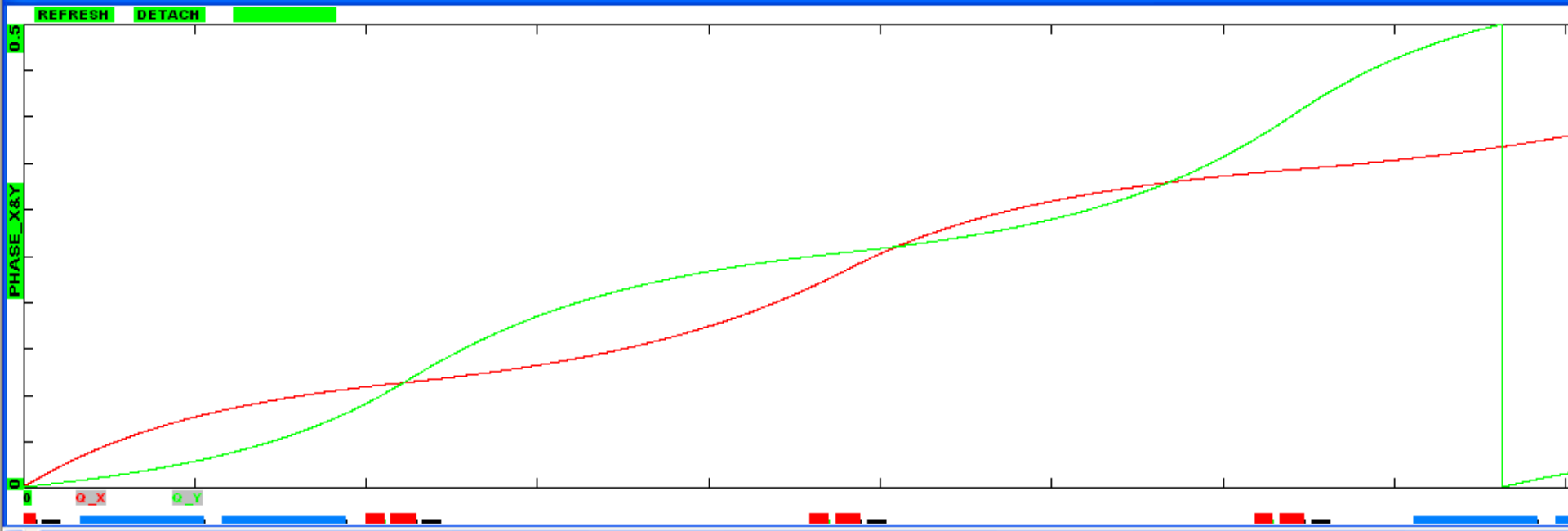
Փոխազդման կետ

OptiM_7

File Edit Search Fitting Tools View View 4D SpaceCharge Window Help



Betatron phases 16

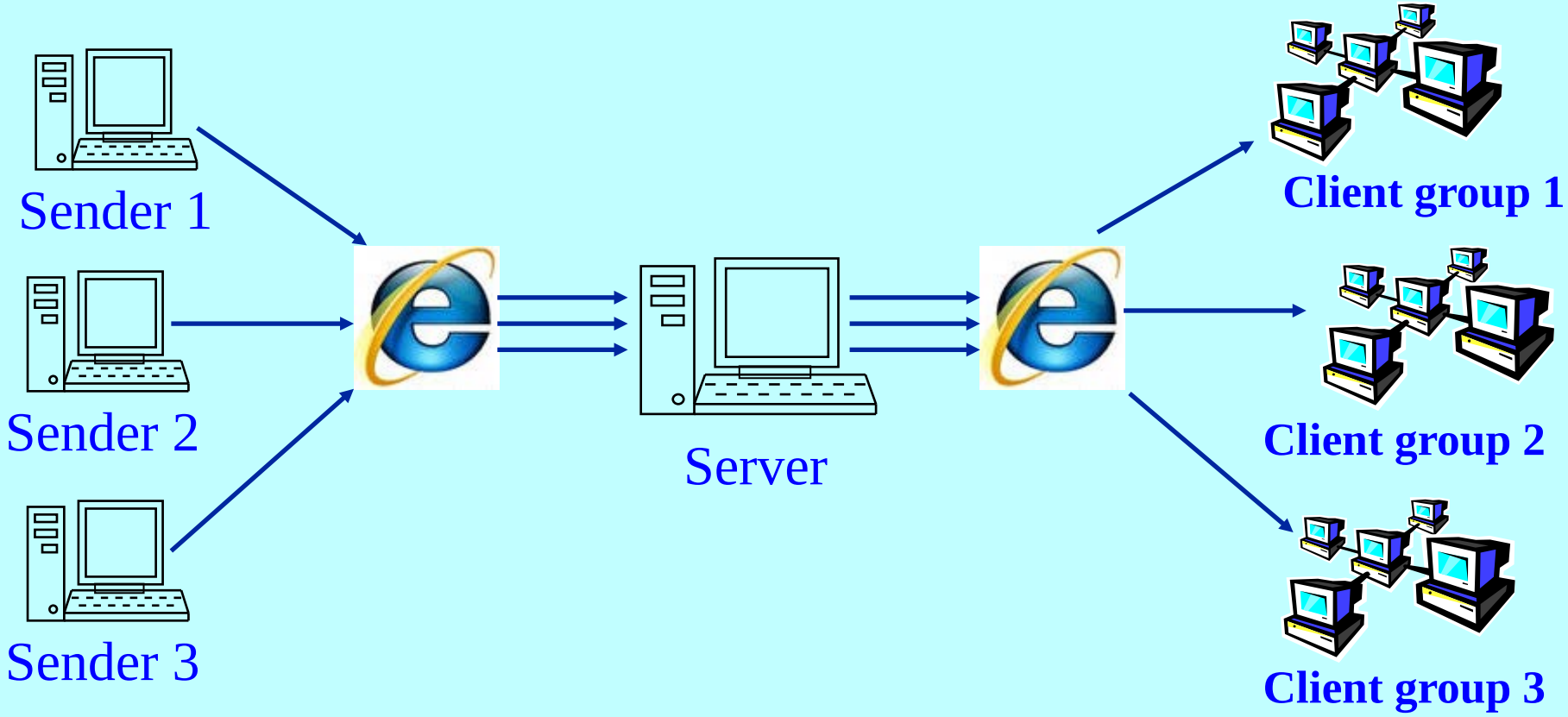


Line 85 of 262

Ի՞նչ է սոկետը

Սոկետը ծրագրային ապահովման անվանում է, որը նախատեսված է պրոցեսների միջև տվյալների փոխանակման համար: Պրոցեսները այս դեպքում կարող են կատարվել ինչպես մեկ համակարգչում, այնպես էլ մի քանի համակարգիչներում, որոնք միացված են ցանցով:

Մի քանի կոմպյուտերներով համատեղ աշխատելու համակարգի սխեման



Ֆունկցիաների ցուցակը

1. bool Connect_Sender(char* a_TaskName, char* a_Server_IP);
2. int WaitForEvent_Sender();
3. int WhichFinished_Sender();
4. int ReceiveBuff_Sender(char* a_pcBuff, int a_nLen);
5. void Disconect_Sender();
6. bool Connect_Client(char* a_TaskName, char* a_Server_IP);
7. int WaitForEvent_Client();
8. int ReceiveBuff_Client(char* a_pcBuff);
9. int SendTaskAnswer_Client(char* a_pcBuff, int a_nBuffLen);
10. int GetClient_ID_InSystem();
11. ….

Կլասների ցուցակը

1. class CASocket;
2. class CASocketDv
3. class CServerDv;
4. class CThreadDv;
5. ….

Բոլոր ֆունկցիաները գտնվում են “SenderAndWorker.dll” գրադարանում: Ֆունկցիաները կարելի է օգտագործել ցանկացած ծրագրից կամ ծրագրավորման լեզվից, որը ունի DLL գրադարան կարդալու հնարավորություն: Օրինակ այս համակարգով կարելի է արագացնել Fortran-ով գրած ծրագրերը (http://www.fortran.bcs.org/2003/porting/part_4/slide_01.html)

program call sum_1d_array_in_testsub_dll ! ... integer :: p pointer :: (q, sum_1d_array) ! this is non-standard ! (but a common extension) integer :: n real, dimension(100) :: array real :: total ! ... p = loadlibrary ("testsub.dll"C) ! the C at the end says - ! add a null byte as in C q = getprocaddress (p, "SUM_1D_ARRAY"C) call sum_1d_array (n, array, total) ! end

Բոլոր կլասները և ֆունկցիաները կարելի է քաշել http://armsoccer.ru կայքից