ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΠΑΝΕΠΙΣΤΗΜΙΟΥ ΙΩΑΝΝΙΝΩΝ
ΠΑΡΑΛΛΗΛΑ ΚΑΙ ΚΑΤΑΝΕΜΗΜΕΝΑ ΣΥΣΤΗΜΑΤΑ
ΓΚΟΓΚΟΣ ΧΡΗΣΤΟΣ, ΓΕΩΡΓΙΟΥ ΓΙΩΡΓΟΣ
ΕΡΓΑΣΙΑ ΜΡΙ
ΑΡΤΑ, ΜΑΙΟΣ 2021

ΕΠΙΛΥΣΗ ΜΕ ΜΡΙ ΤΗΣ ΕΞΙΣΩΣΗΣ ΔΙΑΧΥΣΗΣ ΘΕΡΜΟΤΗΤΑΣ ΤΟΥ POISSON ΣΕ ΜΙΑ ΔΙΑΣΤΑΣΗ

Α. ΤΟ ΠΡΟΒΛΗΜΑ: Ο σειριακός κώδικας ο οποίος σας δίνεται, επιλύει αριθμητικά την εξίσωση διάχυσης θερμότητας του Poisson σε μια διάσταση (κατά μήκος ενός μεταλλικού σύρματος).

Πρόκειται για μια μερική διαφορική εξίσωση συνεχούς χρόνου της μορφής

$$\frac{\partial u(x,t)}{\partial t} = \alpha \frac{\partial^2 u(x,t)}{\partial x^2}$$

με $u(x,t)$ τη θερμοκρασία στη θέση x του σύρματος κατά τη χρονική στιγμή t και α τη σταθερά διάχυσης. Δίνονται ως συνοριακές συνθήκες οι θερμοκρασίες τις οποίες διατηρούμε σταθερές στα δυο άκρα του μεταλλικού σύρματος και αυτό έχει ως συνέπεια, μετά από κάποιο χρονικό διάστημα, να επέλθει μια κατάσταση ισορροπίας με σταθερές θερμοκρασίες σε όλα τα σημεία x του σύρματος. Ο σειριακός κώδικας προσομοιώνει ακριβώς τη διαδικασία διάχυσης της θερμότητας από τα δυο άκρα, μέχρι να επέλθει ισορροπία.

Για να επιλύσουμε την εξίσωση αριθμητικά με υπολογιστή, αφενός διακριτοποιούμε το μήκος του σύρματος με βήμα έστω Δx, καθώς και το χρόνο με βήμα Δt. Το μεταλλικό σύρμα έχει χωριστεί σε ένα σύνολο τμημάτων και η εξέλιξη, με το χρόνο, της θερμοκρασίας σε κάποιο από τα τμήματα αυτά αποδεικνύεται ότι δίνεται από την παρακάτω επαναληπτική σχέση,

$$_i^t u = (1-2F)\ {}_i^{t-1}u + F\ {}_{i-1}^{t-1}u + F\ {}_{i+1}^{t-1}u$$

η οποία επιλύει αριθμητικά την εξίσωση, όπου $_i^t u$ συμβολίζει τη θερμοκρασία του τμήματος i την τρέχουσα χρονική στιγμή t, η οποία με βάση την παραπάνω εξίσωση εξαρτάται από τη θερμοκρασία, κατά την προηγούμενη χρονική στιγμή $t-1$, όχι μόνο του ίδιου του τμήματος στη θέση i, αλλά και των άμεσα γειτονικών του τμημάτων, του $i-1$ στα αριστερά του και του $i+1$ στα δεξιά του. $F = \frac{\alpha \Delta t}{\Delta x^2}$ είναι μια σταθερά χωρίς διαστάσεις, στην οποία ενσωματώνονται όλες οι παράμετροι του προβλήματος. Θεωρούμε ότι το αριστερότερο από τα τμήματα του σύρματος έχει στα αριστερά του ένα εικονικό τμήμα με σταθερή θερμοκρασία ίση με την αρχική συνθήκη θερμοκρασίας στο αριστερό άκρο του σύρματος, και ότι το δεξιότερο τμήμα του σύρματος έχει στα δεξιά του ένα εικονικό τμήμα με σταθερή θερμοκρασία ίση με την αρχική συνθήκη θερμοκρασίας στο δεξιό άκρο του σύρματος.

Β. ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΣΕΙΡΙΑΚΟΥ ΚΩΔΙΚΑ: Στο σειριακό κώδικα, διακριτοποιούμε το μήκος του σύρματος σε `GRIDSIZE` τμήματα, όμως ο πίνακας `u` ο οποίος χρησιμοποιείται για την αποθήκευση των θερμοκρασιών των τμημάτων αυτών κατά τη χρονική στιγμή $t-1$, έχει διάσταση `GRIDSIZE+2`, γιατί στη θέση `u[0]` τοποθετούμε την τιμή της σταθερής θερμοκρασίας στο αριστερό άκρο (αριστερή οριακή συνθήκη) και στη θέση `u[GRIDSIZE+1]` τοποθετούμε την τιμή της σταθερής θερμοκρασίας στο δεξιό άκρο του

σύρματος (δεξιά οριακή συνθήκη). Επομένως, οι θερμοκρασίες των GRIDSIZE τμημάτων του σύρματος βρίσκονται στις θέσεις 1 έως GRIDSIZE του πίνακα u. Αρχικοποιούμε τις θέσεις 1 έως GRIDSIZE του πίνακα u στην τιμή 0, και για παράδειγμα, θέτουμε u[0]=200 και u[GRIDSIZE+1]=5 για να δηλώσουμε ότι η οριακή συνθήκη στο αριστερό άκρο είναι 200 και στο δεξί 5.

Εάν για όλα τα τμήματα εφαρμόσουμε την παραπάνω εξίσωση επαναλαμβανόμενα, συγκλίνουμε στη αριθμητική λύση της εξίσωσης Poisson. Μπορούμε να σταματήσουμε τις επαναλήψεις, όταν η τετραγωνική ρίζα του αθροίσματος των τετραγώνων των διαφορών, για όλα τα τμήματα i, ανάμεσα στη θερμοκρασία τους όπως υπολογίζεται κατά την τρέχουσα χρονική στιγμή (δηλαδή το ${}_i^t u$) και τη θερμοκρασία τους κατά την προηγούμενη χρονική στιγμή (δηλαδή το ${}_i^{t-1} u$), δηλαδή η ποσότητα

$$\sqrt{\sum_i ({}_i^t u - {}_i^{t-1} u)^2}$$

έχει γίνει μικρότερη από την τετραγωνική ρίζα μιας μικρής ποσότητας residual (η απαιτούμενη ακρίβεια σύγκλισης). Στον κώδικα, η unorm ισούται με το τετράγωνο της παραπάνω ποσότητας. Η συνάρτηση poisson_discrete(), αφού υπολογίσει για όλα τα τμήματα i τις νέες θερμοκρασίες τους unew[i] με βάση την προηγούμενη θερμοκρασία τους u[i] και τις προηγούμενες θερμοκρασίες των γειτονικών τους τμημάτων u[i-1] και u[i+1], υπολογίζει και επιστρέφει την ποσότητα unorm. Μέσω της while στη main, επαναλαμβάνουμε τη διαδικασία, όσο η τετραγωνική ρίζα της unorm παραμένει μεγαλύτερη από την τετραγωνική ρίζα του residual. Ένα ιστόγραμμα της λύσης εκτυπώνεται στο τέλος στη γραμμή εντολής.

Γ. ΤΙ ΖΗΤΕΙΤΑΙ: Ζητείται, με χρήση της MPI, να παραλληλοποιήσετε την επαναληπτική αριθμητική λύση της εξίσωσης διάχυσης θερμότητας του Poisson κατά μήκος του σύρματος, έχοντας ως αρχικές συνθήκες τις θερμοκρασίες στα δυο άκρα του και να παρουσιάσετε σε γράφημα μετρήσεις χρόνου εκτέλεσης για διαφορετικούς πληθάριθμους τμημάτων στα οποία διαμερίζουμε το σύρμα και για διαφορετικούς αριθμούς διεργασιών που θα εκτελέσουν τον κώδικα παράλληλα.

Δ. ΠΑΡΑΤΗΡΗΣΕΙΣ ΓΙΑ ΤΟΝ ΠΑΡΑΛΛΗΛΟ ΚΩΔΙΚΑ: Εάν n_ranks είναι ο αριθμός των διεργασιών που χρησιμοποιούμε, κάθε διεργασία θα υπολογίζει τις νέες θερμοκρασίες κατά τη διάρκεια μιας επανάληψης, μόνο για ένα μπλοκ GRIDSIZE/n_ranks διαδοχικών τμημάτων. Κάθε διεργασία θα έχει τα δικά της διανύσματα u και unew όπου θα αποθηκεύει αντίστοιχα τις τιμές θερμοκρασίας κατά τις χρονικές στιγμές $t-1$ και t για τα τμήματα που της έχουν ανατεθεί μόνο. Όμως επειδή η θερμοκρασία κάθε τμήματος χρειάζεται για να την υπολογίσουμε και τις θερμοκρασίες των γειτονικών του τμημάτων, πρέπει η διάσταση των u και unew να είναι (GRIDSIZE/n_ranks)+2. Στη θέση 0 χρειαζόμαστε τη θερμοκρασία του αριστερού γείτονα του αριστερότερου τμήματος που έχει αναλάβει η συγκεκριμένη διεργασία με δείκτη rank, αυτό το έχει όμως διαθέσιμο μόνο η διεργασία rank-1. Παρόμοια, στη θέση (GRIDSIZE/n_ranks)+1 χρειαζόμαστε τη θερμοκρασία του δεξιού γείτονα του δεξιότερου τμήματος που έχει αναλάβει η συγκεκριμένη διεργασία με δείκτη rank, αυτό το έχει όμως διαθέσιμο μόνο η διεργασία rank+1. Έτσι οι διεργασίες πρέπει να ανταλλάσσουν μεταξύ τους τιμές. Σημειώνουμε ότι οι θερμοκρασίες των τμημάτων που έχουν ανατεθεί στη διεργασία rank βρίσκονται στις θέσεις από 1 έως (GRIDSIZE/n_ranks)του πίνακα u της διεργασίας αυτής. Η ανταλλαγή τιμών μεταξύ των διεργασιών πρέπει να γίνει ως εξής: κάθε διεργασία

`rank` (εκτός της διεργασίας `0`) πρέπει να στείλει στη γειτονική της στα αριστερά διεργασία `rank-1` τη θερμοκρασία του αριστερότερου τμήματος που της έχει ανατεθεί και να λάβει από τη διεργασία `rank-1` την τιμή της θερμοκρασίας του δεξιότερου τμήματος που έχει ανατεθεί στη `rank-1`. Παρόμοια, κάθε διεργασία `rank` (εκτός της τελευταίας `n_ranks-1`) πρέπει να στείλει στη γειτονική στα δεξιά της `rank+1` τη θερμοκρασία του τμήματος στο δεξί άκρο της και να λάβει από την `rank+1` τη θερμοκρασία του τμήματος στο αριστερό άκρο της `rank+1`. Για να επιτευχθεί συντονισμός αυτών των ανταλλαγών τιμών στην ανασταλτική επικοινωνία (blocking communication), χωρίζουμε τις διεργασίες με βάση το `rank` σε άρτιες και περιττές. Ενώ οι περιττές διεργασίες στέλνουν πρώτα τις απαραίτητες τιμές στις άρτιες γειτονικές τους διεργασίες και στη συνέχεια λαμβάνουν τιμές από τις άρτιες διεργασίες, αντίθετα οι άρτιες διεργασίες πρώτα θα λαμβάνουν τις τιμές που τους στέλνουν οι γειτονικές τους περιττές διεργασίες και μετά θα στέλνουν δεδομένα στις περιττές γειτονικές τους διεργασίες, τις οποίες αυτές θα περιμένουν.

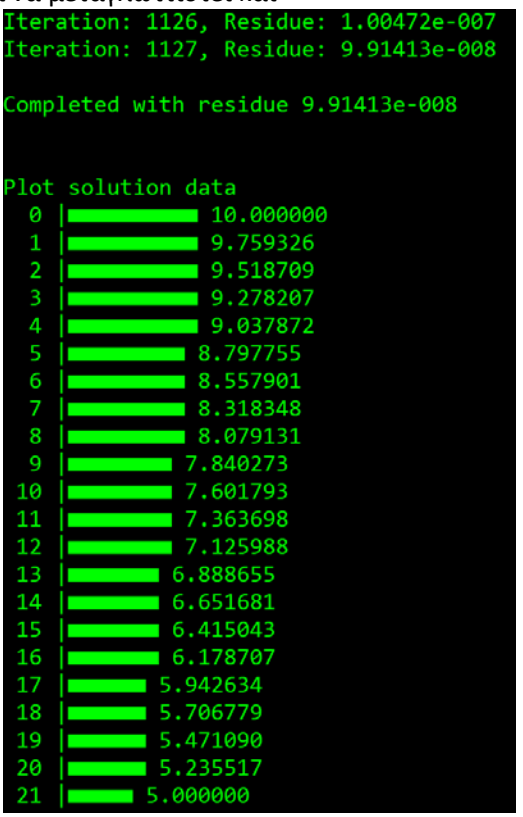
Ο πίνακας 1 που ακολουθεί, θα σας βοηθήσει να συντάξετε τα μηνύματα που πρέπει να ανταλλαγούν μεταξύ των διεργασιών (συναρτήσεις `MPI_Send()` και `MPI_Recv()`), ανάλογα με το εάν κάποια διεργασία έχει περιττό `rank` (1, 3, 5,…), ή άρτιο `rank` (0, 2, 4,…)

Κάθε διεργασία θα υπολογίζει τις νέες τιμές των θερμοκρασιών των τμημάτων που της έχουν ανατεθεί, καθώς και την ποσότητα `unorm` μόνο για το τμήματα τα οποία της έχουν ανατεθεί, ενώ η διεργασία `0` θα αναλάβει να αθροίζει τις επιμέρους τιμές των `unorm` σε μια μεταβλητή `global_unorm` και να ελέγχει εάν πρέπει να προχωρήσουμε, ή όχι, σε επόμενη επανάληψη. Τέλος, όταν έχει εγκατασταθεί ισορροπία στο σύστημα, η διεργασία `0` θα πρέπει να συγκεντρώνει τα επιμέρους αποτελέσματα της λύσης που έχει βρει κάθε διεργασία για τα τμήματα που της έχουν ανατεθεί, για παράδειγμα σε ένα διάνυσμα `uall` και να τα εκτυπώνει.

Ε. ΕΠΙΠΛΕΟΝ ΔΙΕΥΚΡΙΝΙΣΕΙΣ:

1. Μπορεί επιπρόσθετα να χρειαστεί και ο διακόπτης `-lm` κατά τη μεταγλώττιση, ώστε να γίνει η σύνδεση της βιβλιοθήκης `math`. Ο σειριακός κώδικας μπορεί να μεταγλωττιστεί και εκτελεστεί στα Windows με

```
gcc PoissonSerial.c –o run
run
```



Στα Windows, στο τμήμα του κώδικα που εκτυπώνει το ιστόγραμμα, ίσως χρειαστεί να αλλάξετε την `printf("\u2500");` με `printf("%c", (char)254u);` για σωστή εμφάνιση.

Στο Windows Subsystem for Linux, μπορείτε να μεταγλωττίσετε και να τρέξετε με

```
$ gcc PoissonSerial.c –o run -lm
$ ./run
```

αλλά και με

```
$ mpicc PoissonSerial.c –o run –lm
$ mpiexec –n 1 ./run
```

Στο διπλανό σχήμα φαίνεται η έξοδος όταν επιβάλλουμε οριακές

συνθήκες 10 και 5 στα άκρα, με 20 τμήματα και `residual=1e-7;`.

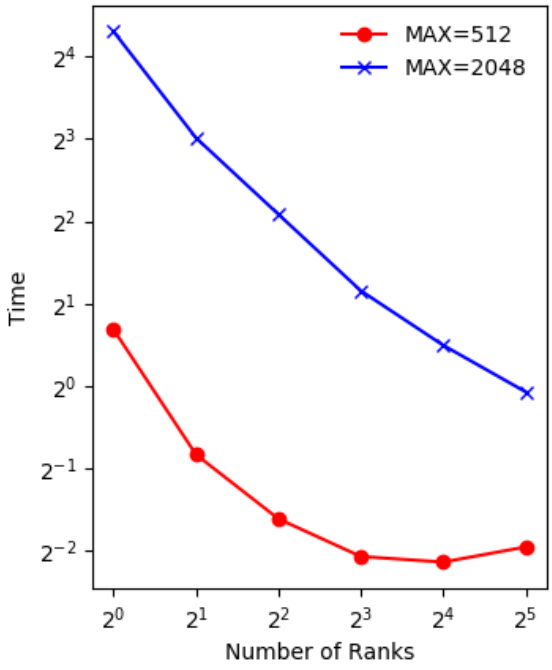
2. Αρχικοποιούμε τη μεταβλητή `unorm` σε μια μεγάλη τιμή όταν ξεκινάνε οι επαναλήψεις.

3. Υποθέτουμε ότι ο αριθμός `GRIDSIZE` των εσωτερικών τμημάτων διαιρείται ακριβώς με τον αριθμό των διεργασιών `n_ranks`.

4. Λάβετε υπόψη, για να ερμηνεύσετε σωστά τα αποτελέσματα που παίρνετε, ότι όταν επιλέγουμε περισσότερα τμήματα (αυξάνοντας το `GRIDSIZE`), ουσιαστικά, επειδή εμείς επιλέγουμε να κρατάμε το βήμα διακριτοποίησης Δx σταθερό στις προσομοιώσεις, αυτό αντιστοιχεί στο να θεωρούμε κάθε φορά κι ένα σύρμα μεγαλύτερο μήκους, ίσου με $l = \Delta x \cdot GRIDSIZE$. Εάν ο σκοπός της προσομοίωσης είναι να διαιρούμε κάθε φορά ένα σύρμα σταθερού μήκους l σε όλο και περισσότερα τμήματα `GRIDSIZE` για μεγαλύτερη ακρίβεια, τότε θα πρέπει κάθε φορά που αλλάζουμε το `GRIDSIZE` να αλλάζουμε παράλληλα και το $\Delta x = \frac{l}{GRIDSIZE}$ στην προσομοίωση. Αυτό θα αλλάζει και την τιμή της παραμέτρου F. Τα προαναφερόμενα σημειώνονται μόνο για την καλύτερη κατανόηση του προβλήματος. Στις δικές μας προσομοιώσεις για μετρήσεις χρόνου εκτέλεσης, επιλέγουμε να κρατάμε το Δx σταθερό και να αυξάνουμε κάθε φορά το `GRIDSIZE`.

5. Το “ΕΝΔΕΧΟΜΕΝΩΣ” που αναφέρεται στον πίνακα 1, αφορά στο γεγονός ότι το τελευταίο rank μπορεί να είναι άρτιο, ή περιττό κι αυτό εξαρτάται από τον αριθμό των διεργασιών που επιλέξαμε.

6. Χρησιμοποιώντας τη συνάρτηση `MPI_Wtime()`, να πειραματιστείτε μετρώντας το χρόνο εκτέλεσης για διαφορετικέ τιμές του `GRIDSIZE` και του αριθμού διεργασιών `n_ranks`. Δοκιμάστε με 256, 512, 1024, 2048 και 4096 τμήματα. Επίσης με 1, 2, 4 και 8 διεργασίες. Κατά τις προσομοιώσεις αυτές να χρησιμοποιήσετε τις τιμές `u[0]=200` και `u[GRIDSIZE+1]=5` για τις οριακές συνθήκες. Προσοχή, να μετρήσετε το χρόνο των επαναλήψεων μόνο, χωρίς να κάνετε εκτυπώσεις κατά τη διάρκεια των επαναλήψεων, αλλά μόνο μετά το πέρας τους, γιατί προσθέτουν σημαντικό overhead στο χρόνο εκτέλεσης. Να παραδώσετε ένα διάγραμμα με τις προσομοιώσεις, όπως το παρακάτω:



ΣΤ. ΠΑΡΑΔΟΣΗ ΕΡΓΑΣΙΑΣ: Θα πρέπει να παραδώσετε ένα αρχείο .zip με το όνομά σας (π.χ. ΓΕΩΡΓΙΟΥ ΓΙΩΡΓΟΣ.zip), το οποίο θα περιλαμβάνει τον κώδικα σε C ο οποίος με MPI υλοποιεί την παραλληλοποίηση, ένα αρχείο κειμένου με όνομα instructions.txt στο οποίο θα υπάρχουν οδηγίες για τη μεταγλώττιση και εκτέλεση του κώδικα, και ένα αρχείο στο οποίο θα περιγράφετε σύντομα ότι διευκρινίσεις θεωρείτε απαραίτητες για τον κώδικά σας, καθώς και τα αποτελέσματα των μετρήσεων των χρόνων εκτέλεσης, σύμφωνα με τις οδηγίες οι οποίες δίνονται παραπάνω. Η εργασία σας θα πρέπει να αναρτηθεί στις Εργασίες του μαθήματος ΚΑΤΑΝΕΜΗΜΕΝΑ ΚΑΙ ΠΑΡΑΛΛΗΛΑ ΣΥΣΤΗΜΑΤΑ του e-class (Εργασία 3 MPI), μέχρι το βράδυ της Παρασκευή 11/6/2021.

ΠΕΡΙΤΤΑ ranks						
	ΣΤΕΛΝΕΙ/ ΛΑΜΒΑΝΕΙ	**ΠΟΙΟΣ ΣΤΕΛΝΕΙ/ ΠΟΙΟΣ ΛΑΜΒΑΝΕΙ**	**ΤΙ ΣΤΕΛΝΕΙ/ΤΙ ΛΑΜΒΑΝΕΙ**	**ΣΕ ΠΟΙΟΝ ΣΤΕΛΝΕΙ/ ΑΠΟ ΠΟΙΟΝ ΛΑΜΒΑΝΕΙ**	**ΠΟΥ ΘΑ ΑΠΟΘΗΚΕΥΤΕΙ/ ΑΠΟ ΠΟΥ ΠΡΟΕΡΧΕΤΑΙ**	**TAG**
A1	ΣΤΕΛΝΕΙ	ΟΛΑ ΤΑ ΠΕΡΙΤΤΑ RANKS (`rank`)	`u[1]`	ΣΤΟ ΑΡΤΙΟ RANK (`rank-1`)	`u[points+1]`	1
B2	ΛΑΜΒΑΝΕΙ	ΟΛΑ ΤΑ ΠΕΡΙΤΤΑ RANKS (`rank`)	`u[points]`	ΑΠΟ ΤΟ ΑΡΤΙΟ RANK (`rank-1`)	`u[0]`	2
C1	ΣΤΕΛΝΕΙ	ΟΛΑ ΤΑ ΠΕΡΙΤΤΑ RANKS (`rank`) ΕΚΤΟΣ ΕΝΔΕΧΟΜΕΝΩΣ ΤΟΥ `n_ranks-1`	`u[points]`	ΣΤΟ ΑΡΤΙΟ RANK (`rank+1`)	`u[0]`	1
D2	ΛΑΜΒΑΝΕΙ	ΟΛΑ ΤΑ ΠΕΡΙΤΤΑ RANKS (`rank`) ΕΚΤΟΣ ΕΝΔΕΧΟΜΕΝΩΣ ΤΟΥ `n_ranks-1`	`u[1]`	ΑΠΟ ΤΟ ΑΡΤΙΟ RANK (`rank+1`)	`u[points+1]`	2
ΑΡΤΙΑ ranks						
	ΣΤΕΛΝΕΙ/ ΛΑΜΒΑΝΕΙ	**ΠΟΙΟΣ ΣΤΕΛΝΕΙ/ ΛΑΜΒΑΝΕΙ ΛΑΜΒΑΝΕΙ**	**ΤΙ ΣΤΕΛΝΕΙ/ΤΙ ΛΑΜΒΑΝΕΙ**	**ΣΕ ΠΟΙΟΝ ΣΤΕΛΝΕΙ/ ΑΠΟ ΠΟΙΟΝ ΛΑΜΒΑΝΕΙ**	**ΠΟΥ ΘΑ ΑΠΟΘΗΚΕΥΤΕΙ/ ΑΠΟ ΠΟΥ ΠΡΟΕΡΧΕΤΑΙ**	**TAG**
A2	ΛΑΜΒΑΝΕΙ	ΟΛΑ ΤΑ ΑΡΤΙΑ RANKS (`rank`) ΕΚΤΟΣ ΕΝΔΕΧΟΜΕΝΩΣ ΤΟΥ `n_ranks-1`	`u[1]`	ΑΠΟ ΤΟ ΠΕΡΙΤΤΟ RANK (`rank+1`)	`u[points+1]`	1
B1	ΣΤΕΛΝΕΙ	ΟΛΑ ΤΑ ΑΡΤΙΑ RANKS (`rank`) ΕΚΤΟΣ ΕΝΔΕΧΟΜΕΝΩΣ ΤΟΥ `n_ranks-1`	`u[points]`	ΣΤΟ ΠΕΡΙΤΤΟ RANK (`rank+1`)	`u[0]`	2
C2	ΛΑΜΒΑΝΕΙ	ΟΛΑ ΤΑ ΑΡΤΙΑ RANKS (`rank`) ΕΚΤΟΣ ΤΟΥ 0	`u[points]`	ΑΠΟ ΤΟ ΠΕΡΙΤΤΟ RANK (`rank-1`)	`u[0]`	1
D1	ΣΤΕΛΝΕΙ	ΟΛΑ ΤΑ ΑΡΤΙΑ RANKS (`rank`) ΕΚΤΟΣ ΤΟΥ 0	`u[1]`	ΣΤΟ ΠΕΡΙΤΤΟ RANK (`rank-1`)	`u[points+1]`	2

Πίνακας 1. Μηνύματα που ανταλλάσσονται μεταξύ άρτιων και περιττών ranks. Ίδια χρώματα δηλώνουν μηνύματα που κάποιο rank (άρτιο ή περιττό) στέλνει/λαμβάνει και κάποιο αντίστοιχο rank (περιττό, ή άρτιο) λαμβάνει/στέλνει