ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

Συστήματα Παράλληλης και Κατανεμημένης Επεξεργασίας

Ενότητα: ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ No:18 **HellasGrid -Advanced**

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http://arch.icte.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα του Πανεπιστημίου Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα

1. Σκοπός της άσκησης

- Προχωρημένα θέματα HellasGrid.
- Περισσότερα για proxy και αρχεία JDL.
- Διαχείριση αποθήκευσης δεδομένων.
- Χρήση προγράμματος icc και της βιβλιοθήκης MKL.
- Μέτρηση επιδόσεων.

2. Παραδοτέα

(A) 14 ερωτήσεις

(C) 6 ασκήσεις

3. Proxy – Advanced info

(A1) Τι ισχύ έχει ένα πιστοποιητικό proxy στο HellasGrid;

Πολλές φορές μπορεί να χρειαστεί να δημιουργήσετε ένα proxy με μεγαλύτερη διάρκεια για τη δική σας διευκόλυνση. Γι' αυτό υπάρχει η εντολή

```
myproxy-init -n -d -R 'wms*.egee-see.org|wms*.hellasgrid.gr'
```

Εκτελέστε την και απαντήστε στα εξής:

(A2) Ποια είναι η ισχύς του proxy σας τώρα;

(A3) Με ποια εντολή μπορείτε να το επιβεβαιώσετε;

(A4) Υπάρχει κάποια παράμετρος που να μπορεί να καθορίσει πόσο χρόνο ισχύς θα έχει το πιστοποιητικό;

(A5) Θέστε το πιστοποιητικό σε ισχύ για 10 μέρες. Επιβεβαιώστε με screenshot.

(A6) Ποιά είναι η μέγιστη ισχύς που μπορεί να έχει το πιστοποιητικό;

(A7) Ποια είναι η εντολή διαγραφής του proxy;

Αξίζει να σημειωθεί ότι παρόλο που μπορούμε να θέσουμε την ισχύ του πιστοποιητικού για πολύ καιρό, οι εργασίες που υποβάλλονται πρέπει να έχουν ολοκληρωθεί πριν από το **WallClockTime** όριο, το οποίο είναι συνήθως από 3 έως 10 μέρες. Εξαρτάται από το CE. Εάν υπερβεί αυτή η χρονική περίοδος η εργασία θα ακυρωθεί.

4. JDL αρχεία – Advanced info

Όπως έχουμε ήδη δει κάθε εργασία που υποβάλλεται στο Grid περιγράφεται από ένα JDL αρχείο που σαν ονομασία προέρχεται από τα αρχικά των λέξεων Job Description Language. Ένα JDL αρχείο είναι ένα απλό αρχείο κειμένου *(ASCII text)* με κατάληξη .jdl και περιλαμβάνει μία σειρά παραμέτρων, που περιγράφουν την εργασία, και τις τιμές αυτών.

Για την γρήγορη σύνταξη αρχείων JDL η ομάδα ανάπτυξης υπηρεσιών έχει αναπτύξει το εργαλείο jdl-creator.

(A8) Εντοπίστε και αντιγράψτε το εργαλείο jdl-creator στον κατάλογο εργασίας.

Από τις διαθέσιμες επιλογές ο χρήστης μπορεί να επιλέξει πολλαπλές είτε διαχωρίζοντας με κόμμα `(,)` τις επιλογές του είτε ορίζοντας ένα εύρος τιμών με παύλα `(-)`. Επιπλέον δίνεται η δυνατότητα να χρησιμοποιηθεί η τιμή `'0'` που αυτομάτως θα συμπεριλάβει στο πρότυπο JDL τα απαραίτητα πεδία που πρέπει να έχει το τελικό αρχείο που θα χρησιμοποιηθεί.

Ανάλογα με τις ανάγκες που θέλουμε να πληρεί η εργασία μας μπορούμε να επιλέξουμε ανάμεσα στις διαθέσιμες επιλογές ή να κάνουμε συνδυασμό αυτών ώστε στο τέλος να έχουμε ένα πρότυπο JDL ή έστω ένα τμήμα αυτού για να χρησιμοποιήσουμε ως βάση. Την έξοδο λοιπόν από το εργαλείο ο χρήστης μπορεί να την αντιγράψει με copy paste και να δημιουργήσει έτσι εύκολα και γρήγορα το πλήρες JDL αρχείο παραμετροποιημένο για τη δική του εργασία.

Επιπλέον των παραπάνω το εργαλείο δίνει τη δυνατότητα άμεσης εγγραφής των πρότυπων γραμμών σε νέο αρχείο εφόσον κληθεί με την παρακάτω σύνταξη:

```
$jdl-creator -o job.jdl
```

Τέλος εάν βρισκόμαστε εντός του καταλόγου όπου υπάρχουν το εκτελέσιμο και τα input δεδομένα μπορούμε να καλέσουμε το εργαλείο με την ακόλουθη σύνταξη:

```
$jdl-creator -s -m -o job.jdl
```

και αυτό θα προσπαθήσει να ταυτοποιήσει ποιο είναι το εκτελέσιμο και ποια τα αρχεία εισόδου ώστε να τα συμπεριλάβει απευθείας στις αντίστοιχες γραμμές *(Executable,InputSandbox)*.

Η εντολή χρήσης αυτού του εργαλείου είναι:

```
jdl-creator [options] [arguments]
```

και οι επιλογές είναι οι εξής:

`--version`	Εμφάνιση της έκδοσης και τερματισμός
`-h, --help`	Εμφάνιση βοήθειας και τερματισμός
`-o FILE, --output=FILE`	Αποθήκευση της εξόδου στο FILE

`-m, --make-jdl`	Κάνει την έξοδο ολοκληρωμένο JDL
`-a field/s, --arguments=field/s`	Διαθέσιμες επιλογές
`-f, --force`	Αναγκάζει την έξοδο, ακόμη και αν υπάρχουν συγκρούσεις
`-q, --quiet`	Απόκρυψη εξόδου
`-s, --smart`	Προσπάθεια αναγνώρισης του Executable και InputSandbox
`-l, --list`	Εμφάνιση όλων των διαθέσιμων πεδίων
`-e, --examples`	Εμφάνιση κάποιων παραδειγμάτων χρήσης

(A9) Επιτύχετε το παρακάτω αποτέλεσμα μέσω του εργαλείου jdl-creator.

```
# ----------------------------------------
# JDL Template Created by jdl-creator.
# More info @(support@grid.auth.gr)
# ----------------------------------------
Executable = "run.sh";
CpuNumber = 8;
StdOutput = "std.out";
StdError = "std.err";
InputSandbox = {"run.sh", "data.dat"};
OutputSandbox = {"std.err","std.out"};
Environment = {"NODES_REQ=1:cuda"};
Requirements = Member("OPENMPI",
other.GlueHostApplicationSoftwareRunTimeEnvironment)
&& Member("CUDA",
other.GlueHostApplicationSoftwareRunTimeEnvironment);
```

Ποιά εντολή δώσατε και ποιες επιλογές επιλέξατε μέσα στο εργαλείο, ώστε να δημιουργηθεί το αρχείο example.jdl με τα παραπάνω περιεχόμενα;

5. Διαχείριση αποθήκευσης δεδομένων στο HellasGrid

Το HellasGrid δίνει τη δυνατότητα στους χρήστες του να διαχειρίζονται μεγάλου όγκου αρχεία, εισόδου και εξόδου, και τη μεταφορά τους σε όλα τα αποθηκευτικά μηχανήματα της υποδομής. Όταν κανείς θέλει να αποθηκεύσει τα αρχεία του ή να τα συγκεντρώσει σε ένα σύστημα πρέπει να ακολουθήσει μία συγκεκριμένη διαδικασία, η οποία θα αναλυθεί σε λίγο.

Αρχικά, ας αναφέρουμε μερικά πράγματα σχετικά με τον τρόπο αποθήκευσης των αρχείων στο grid. Τα αρχεία που αποθηκεύονται στους αποθηκευτικούς χώρους του HellasGrid, μπορούμε να τα παρακολουθούμε με το Logical File Catalog *(LFC)*. Κάθε αρχείο που αποθηκεύετε για πρώτη φορά στους αποθηκευτικούς χώρους, καταχωρείτε με κάποιο όνομα στο LFC. Μπορούμε να αναφερθούμε στα αρχεία με πολλά ονόματα, τα πιο συνηθισμένα όμως είναι το GUID (Grid Unique Identifier), το

LFN *(Logical File Name)* και το SURL *(Storage Uniform Resource Locator ή Storage URL)*. Μπορεί να έχει πάνω από ένα λογικό όνομα αλλά έχει μόνο ένα GUID.

- Το GUID αναφέρεται σε ένα συγκεκριμένο αρχείο καθώς και σε όλα τα αντίγραφά του στα storage elements.

- Επειδή είναι δύσκολη η διαχείριση των αρχείων με το GUID όνομά τους, χρησιμοποιούμε το LFN, το οποίο είναι πιο εύκολο στη χρήση και την κατανόηση. Μπορούμε να αναφερόμαστε στο ίδιο αρχείο με περισσότερα από ένα LFN ονόματα, ασχέτως με το πόσα αντίγραφά του υπάρχουν στην υποδομή.

- Το SURL δείχνει την ακριβή διεύθυνση του αρχείου στο grid. Γι' αυτό για κάθε αντίγραφο του αρχείου, υπάρχει ένα μοναδικό SURL που το χαρακτηρίζει. Δεν είναι πρακτική η χρήση του SURL και γι' αυτό αποφεύγεται.

Όποιο από τα παραπάνω αναγνωριστικά και αν χρησιμοποιείτε για να αναφερθείτε σε ένα αρχείο, αυτό επιτυγχάνεται με την υπηρεσία του Logical File Catalog. Υπάρχουν διαθέσιμες εντολές για το 'ανέβασμα' και 'κατέβασμα' των αρχείων στα και από τα storage elements, οι λεγόμενες lcg-εντολές.

Πριν εκτελέσετε αυτές τις εντολές πρέπει να ρυθμιστούν κάποιες μεταβλητές περιβάλλοντος.

Εκτελέστε με τη σειρά τις παρακάτω εντολές:

```
export LCG_CATALOG_TYPE=lfc

export LFC_HOST=lfc.isabella.grnet.gr

export LCG_GFAL_INFOSYS=bdii.isabella.grnet.gr:2170

export LCG_GFAL_VO=see
```

Μετά την εκτέλεση των παραπάνω εντολών, για να χρησιμοποιήσετε τις εντολές lcg, θα πρέπει να δημιουργήσετε πιστοποιητικό proxy(εάν δεν έχετε ήδη δημιουργήσει πιο πριν).

Εκτελέστε την εντολή:

```
lcg-cr -l /grid/see/NAME/DIR/FILE file:$PWD/FILE
```

και μεταφέρεται ένα αρχείο της επιλογής σας στο Grid.

(A10) Επιβεβαιώστε με screenshot ότι όλα εκτελέστηκαν σωστά.

Οι εντολές που διατίθονται για τη διαχείριση των storeage elements είναι οι lfc- .

Command	Description	Example	Explanation
lfc-mkdir	Δημιουργεί ένα κατάλογο εργασίας	`lfc-mkdir /grid/see/jsmith/my_project`	Δημιουργία καταλόγου `my_project` στο `/grid/see/jsmith`
lfc-ls	Εμφάνιση περιεχομένων ενός καταλόγου	`lfc-ls -l /grid/see/jsmith`	Επιστρέφει τα περιεχόμενα του καταλόγου.
lfc-rm	Διαγραφή ενός αρχείου/καταλόγου	`lfc-rm -r /grid/see/jsmith/my_project`	Διαγράφει το `my_project` *(πρέπει να είναι άδειο)*
lfc-chmod	Αλλαγή δικαιωμάτων	`lfc-chmod 700 /grid/see/jsmith/my_project`	Θέτει δικαιώματα στον κατάλογο `my_project.`
lfc-rename	Μετονομασία ενός αρχείου/καταλόγου	`lfc-rename /grid/see/jsmith/my_data /grid/see/jsmith/data`	Αλλάζει το όνομα από `my_data` σε `data.`

Οι παραπάνω εντολές χρησιμοποιούν το LFN.

Για τη διαχείριση των αρχείων στο Grid, χρησιμοποιούνται οι εντολές lcg- .

Οι οποίες βρίσκονται στον επόμενο πίνακα.

Command	Description	Example	Explanation
lcg-cr	Αντιγραφή ενός τοπικού αρχείου στον SE (ανέβασμα).	`lcg-cr -l /grid/see/jsmith/my_project/input.tar.gz file:$PWD/input.tar.gz`	Ανεβάζει το αρχείο `input.tar.gz.` Επίσης καταχωρεί αυτό το αρχείο στο LFC με το όνομα `/grid/see/jsmith/my_project/input.tar.gz`
lcg-rep	Αντιγραφή αρχείου	`lcg-rep lfn:/grid/see/jsmith/my_project/input.tar.gz`	Αντιγράφει το αρχείο με το συγκεκριμένο LFN, σε ένα άλλο storage element.
lcg-cp	Αντιγραφή ενός αρχείου του grid στον τοπικό κατάλογο (κατέβασμα).	`lcg-cp lfn:/grid/see/jsmith/my_project/input.tar.gz file:$PWD/input.tar.gz`	Κατέβασμα του αρχείου `input.tar.gz` στο UI και αποθήκευσή του με το όνομα `input.tar.gz`
lcg-del	Διαγραφή των αντιγράφων.	`lcg-del -a lfn:/grid/see/jsmith/my_project/input.tar.gz`	Διαγράφει όλες τις εμφανίσεις του αρχείου στο grid.
lcg-lg	Εμφάνιση του GUID ενός αρχείου.	`lcg-lg lfn:/grid/see/jsmith/my_project/input.tar.gz`	Επιστρέφει το GUID όνομα του αρχείου.
lcg-lr	Εμφάνιση των SURL(s) ενός αρχείου.	`lcg-lr lfn:/grid/see/jsmith/my_project/input.tar.gz`	Επιστρέφει τα GUID(s) ονόματα του αρχείου.

(C1) Δημιουργήστε ένα νέο φάκελο στο LFC και κάντε τον ιδιωτικό, ώστε να έχετε μόνο εσείς πρόσβαση. Αν υποθέσουμε ότι θέλουμε να ανεβάσουμε το αρχείο με όνομα **`local_data.txt`** στο grid στον κατάλογο που δημιουργήσαμε πριν, πώς θα βρούμε το GUID του;

(C2) Αντιγράψτε τώρα το παραπάνω αρχείο στο **se01.athena.hellasgrid.gr**. Δείτε όλα τα διαθέσιμα αντίγραφα που υπάρχουν στην υποδομή και στη συνέχεια διαγράψτε τα.

6. Χρήση Επεξεργαστή icc και της βιβλιοθήκης MKL

Στο HellasGrid υπάρχει η δυνατότητα της χρήσης του compiler της Intel icc. Για τη χρήση του πρέπει να ρυθμιστούν οι κατάλληλες μεταβλητές περιβάλλοντος. Φορτώστε την 11η έκδοση της intel με την εντολή **module load intel/11.1** . Επιβεβαιώστε ότι φορτώθηκε με **module list**. Το αποτέλεσμα πρέπει να είναι:

Currently Loaded Modulefiles:

1) **intel/11.1**

Δοκιμάστε να εκτελέσετε την εντολή **icc -V**.

Εάν σας εμφανίζει το παρακάτω σφάλμα πρέπει να αποκτήσετε ένα license.

```
[user@ui:~]$ icc -V

Error: A license for CCompL is not available (-76,61026,2).

License file(s) used were (in this order):
1. Trusted Storage
2. /home/lmount/.intel/licenses
3. /opt/intel/Compiler/11.1/059/Licenses
4. /home/lmount/intel/licenses
5. /opt/intel/licenses/*.lic
6. /Users/Shared/Library/Application Support/Intel/Licenses
7. /opt/intel/Compiler/11.1/059/bin/intel64/*.lic

Please visit http://support.intel.com/support/performancetools/support.htm
if you require technical assistance.

icc: error #10052: could not checkout FLEXlm license
```

Για να το αποκτήσετε επισκεφθείτε τη σελίδα http://software.intel.com/en-us/non-commercial-software-development επιλέξτε το **intel Math Kernel Library for Linux** και συμπληρώστε στη συνέχεια τη φόρμα που θα σας παρουσιαστεί βάζοντας:

Version	10.0 - 10.2
OS	Linux
Processor Archtecture	Intel(R) 64
Dynamic or Static	Static
Integer length	32-bit (lp64)
Sequential or Multi-threaded	Sequential

Έπειτα στο χώρο του UI *(Users Interface)*, αντιγράψτε το license *(.lic)* σε έναν από τους δύο φακέλους:

1. **~/intel/licenses/**

2. **~/.intel/licenses/**

Εάν δεν υπάρχουν, δημιουργείστε τους με **mkdir**.

(A11) Ξαναδώστε icc –V και επιβεβαιώστε ότι έχετε ολοκληρώσει τη διαδικασία με επιτυχία.

Η Intel έχει αναπτύξει κάποιες βιβλιοθήκες τις λεγόμενες mkl, οι οποίες αποτελούν εναλλακτικές υλοποιήσεις των δημοφιλών βιβλιοθηκών γραμμικής άλγεβρας *(blas, lapack, blacs, scalapack)* βελτιστοποιημένες για την αρχιτεκτονική των επεξεργαστών της Intel.

Θα ακολουθήσουμε τώρα το εργαστηριακό φυλλάδιο της Intel για το mkl. Μπορείτε να το βρείτε εδώ.

Αυτό το tutorial δείχνει πώς να χρησιμοποιούμε τους επεξεργαστές της Intel MKL στις εφαρμογές μας:

- Πολλαπλασιάζοντας πίνακες με τη χρήση των Intel MKL ρουτινών.
- Τη μέτρηση της απόδοσης του πολλαπλασιασμού πινάκων.
- Έλεγχος threading.

Η Intel MKL παρέχει αρκετές ρουτίνες για τον πολλαπλασιασμό πινάκων. Η πιο διαδεδομένη είναι η dgemm ρουτίνα, η οποία υπολογίζει το προϊόν των διπλής ακρίβειας πινάκων:

C = αA*B+β*C

Η dgemm ρουτίνα μπορεί να εκτελέσει πολλούς υπολογισμούς. Για παράδειγμα, μπορείτε να εκτελέσετε αυτή τη λειτουργία με αντιστροφή πινάκων *(transpose)* είτε με συζυγή αντστροφή *(conjugate transpose).*

Οι πλήρεις λεπτομέρειες των δυνατοτήτων της dgemm ρουτίνας και του σύνολο των παραμέτρων της, μπορούν να βρεθούν στο ?gemm θέμα της Intel Math Kernel Manual Reference Library.

Οι A,B πίνακες περιέχουν τα εξής στοιχεία και οι μεταβλητές α, β είναι διπλής ακρίβειας:

$$A = \begin{bmatrix} 1.0 & 2.0 & 3.0 & \cdots & 1000.0 \\ 1001.0 & 1002.0 & 1003.0 & \cdots & 2000.0 \\ 2001.0 & 2002.0 & 2003.0 & \cdots & 3000.0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 999001.0 & 999002.0 & 999003.0 & \cdots & 1000000.0 \end{bmatrix} \quad B = \begin{bmatrix} -1.0 & -2.0 & -3.0 & \cdots & -1000.0 \\ -1001.0 & -1002.0 & -1003.0 & \cdots & -2000.0 \\ -2001.0 & -2002.0 & -2003.0 & \cdots & -3000.0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -999001.0 & -999002.0 & -999003.0 & \cdots & -1000000.0 \end{bmatrix}$$

Η άσκηση αυτή δείχνει πώς μπορείτε να καλέσετε τον dgemm ρουτίνα. Η συγκεκριμένη εφαρμογή θα κάνει χρήση του αποτελέσματος του πολλαπλασιασμού των δύο πινάκων.

Αντιγράψτε τον παρακάτω κώδικα στο αρχείο **dgemm_example.c** και κάντε τον compile με **icc -mkl dgemm_example.c** .

```
#define min(x,y) (((x) < (y)) ? (x) : (y))
#include <stdio.h>
#include <stdlib.h>
#include "mkl.h"
int main()
{
double *A, *B, *C;
int m, n, k, i, j;
double alpha, beta;
printf ("\n This example computes real matrix C=alpha*A*B+beta*C using \n"
" Intel(R) MKL function dgemm, where A, B, and C are matrices and \n"
" alpha and beta are double precision scalars\n\n");
m = 2000, k = 200, n = 1000;
printf (" Initializing data for matrix multiplication C=A*B for matrix \n"
" A(%ix%i) and matrix B(%ix%i)\n\n", m, k, k, n);
alpha = 1.0; beta = 0.0;
printf (" Allocating memory for matrices aligned on 64-byte boundary for
better \n"
" performance \n\n");
A = (double *)mkl_malloc( m*k*sizeof( double ), 64 );
B = (double *)mkl_malloc( k*n*sizeof( double ), 64 );
C = (double *)mkl_malloc( m*n*sizeof( double ), 64 );
if (A == NULL || B == NULL || C == NULL) {
printf( "\n ERROR: Can't allocate memory for matrices. Aborting... \n\n");
mkl_free(A);
mkl_free(B);
mkl_free(C);
return 1;
}
printf (" Intializing matrix data \n\n");
for (i = 0; i < (m*k); i++) {
A[i] = (double)(i+1);
}
for (i = 0; i < (k*n); i++) {
B[i] = (double)(-i-1);
}
for (i = 0; i < (m*n); i++) {
C[i] = 0.0;
}
printf (" Computing matrix product using Intel(R) MKL dgemm function via
CBLAS interface \n\n");
cblas_dgemm(CblasRowMajor, CblasNoTrans, CblasNoTrans,
m, n, k, alpha, A, k, B, n, beta, C, n);
printf ("\n Computations completed.\n\n");
```

```
printf (" Top left corner of matrix A: \n");
for (i=0; i<min(m,6); i++) {
for (j=0; j<min(k,6); j++) {
printf ("%12.0f", A[j+i*k]);
}
printf ("\n");
}
printf ("\n Top left corner of matrix B: \n");
for (i=0; i<min(k,6); i++) {
for (j=0; j<min(n,6); j++) {
printf ("%12.0f", B[j+i*n]);
}
printf ("\n");
}
printf ("\n Top left corner of matrix C: \n");
for (i=0; i<min(m,6); i++) {
for (j=0; j<min(n,6); j++) {
printf ("%12.5G", C[j+i*n]);
}
printf ("\n");
}
printf ("\n Deallocating memory \n\n");
mkl_free(A);
mkl_free(B);
mkl_free(C);
printf (" Example completed. \n\n");
return 0;}
```

(C3) Εάν έγινε σωστά θα πρέπει να δημιουργήθηκε το **a.out** εκτελέσιμο. Εκτελέστε το και δείτε το αποτέλεσμα της πράξης.

7. Μέτρηση επιδόσεων

Η Intel MKL παρέχει λειτουργίες για τη μέτρηση των επιδόσεων. Αυτό παρέχει έναν τρόπο για την ποσοτικοποίηση της απόδοσης βελτίωσης που θα προέκυπτε από τη χρήση Intel MKL ρουτίνες σε αυτό το εγχειρίδιο.

Χρησιμοποιήστε τη **dsecnd** ρουτίνα, για να επιστρέψει το χρόνο που πέρασε στη CPU σε δευτερόλεπτα.

(C4) Αντιγράψτε το **dgemm_example.c** στο **dgemm_example_time.c**. Προσθέστε στο κατάλληλο σημείο το παρακάτω κομμάτι κώδικα και μετρήστε τον χρόνο εκτέλεσης του προγράμματος. Επειδή εκτελείτε πολύ γρήγορα η dgemm, και είναι δύσκολη η μέτρησή της, την εκτελούμε πολλές φορές και μετράμε τον μέσο χρόνο εκτέλεσης. Γι'αυτό το λόγο θέστε μία σταθερή μεταβλητή LOOP_COUNT που θα καθορίζει πόσες φορές θα εκτελεστεί η εντολή.

Για να βρούμε τη βελτιστοποίηση που επιτυγχάνεται με τη χρήση της dgemm, πρέπει να εκτελέσουμε την ίδια μέτρηση, αλλά αντί αυτής της συνάρτησης να υλοποιήσουμε μία τριπλή επανάληψη υπολογισμού των πινάκων.

(C5) Αντιγράψτε το **dgemm_example_time.c** στο **dgemm_example_time2.c**. Για τη χρονομέτρηση της εκτέλεσης, χρησιμοποιείστε τον παρακάτω κώδικα.

```
printf (" Making the first run of matrix product using triple nested loop\n"
" to get stable run time measurements \n\n");
for (i = 0; i < m; i++) {
for (j = 0; j < n; j++) {
sum = 0.0;
for (l = 0; l < k; l++)
sum += A[k*i+l] * B[n*l+j];
C[n*i+j] = sum;
}
}
printf (" Measuring performance of matrix product using triple nested loop
\n\n");
s_initial = dsecnd();
for (r = 0; r < LOOP_COUNT; r++) {
for (i = 0; i < m; i++) {
for (j = 0; j < n; j++) {
sum = 0.0;
for (l = 0; l < k; l++)
sum += A[k*i+l] * B[n*l+j];
C[n*i+j] = sum;
}
}
}
s_elapsed = (dsecnd() - s_initial) / LOOP_COUNT;
printf (" == Matrix multiplication using triple nested loop completed == \n"
" == at %.5f milliseconds == \n\n", (s_elapsed * 1000));
```

(A12) Βρείτε το συνολικό speedup.

8. Μέτρηση της επίδρασης της νημάτωσης στην dgemm

Από προεπιλογή, η Intel MKL χρησιμοποιεί n νήματα, όπου n είναι ο αριθμός των φυσικών πυρήνων στο σύστημα. Περιορίζοντας τον αριθμό των νημάτων και μετρώντας την αλλαγή στην απόδοση της dgemm, η άσκηση αυτή δείχνει πώς το threading επιδρά στην απόδοση του προγράμματος.

Η άσκηση αυτή κάνει χρήση της mkl_set_num_threads ρουτίνας για να γράψει πάνω στον προκαθορισμένο αριθμό νημάτων και της ρουτίνας mkl_get_max_threads, για να καθορίσει τον μέγιστο αριθμό νημάτων.

(C6) Αντιγράψτε το **dgemm_example.c** στο **dgemm_example_threads.c**. Αφού προσθέσετε τον παρακάτω κώδικα με όποιες αλλαγές χρειάζεται, συγκρίνεται το χρόνο εκτέλεσης με τον αρχικό.

```
printf (" Finding max number of threads Intel(R) MKL can use for parallel
runs \n\n");
max_threads = mkl_get_max_threads();
printf (" Running Intel(R) MKL from 1 to %i threads \n\n", max_threads);
for (i = 1; i <= max_threads; i++) {
for (j = 0; j < (m*n); j++)
C[j] = 0.0;
```

```
printf (" Requesting Intel(R) MKL to use %i thread(s) \n\n", i);
mkl_set_num_threads(i);
printf (" Making the first run of matrix product using Intel(R) MKL dgemm
function \n"
" via CBLAS interface to get stable run time measurements \n\n");
cblas_dgemm(CblasRowMajor, CblasNoTrans, CblasNoTrans,
m, n, k, alpha, A, k, B, n, beta, C, n);
printf (" Measuring performance of matrix product using Intel(R) MKL dgemm
function \n"
" via CBLAS interface on %i thread(s) \n\n", i);
s_initial = dsecnd();
for (r = 0; r < LOOP_COUNT; r++) {
cblas_dgemm(CblasRowMajor, CblasNoTrans, CblasNoTrans,
m, n, k, alpha, A, k, B, n, beta, C, n);
}
s_elapsed = (dsecnd() - s_initial) / LOOP_COUNT;
printf (" == Matrix multiplication using Intel(R) MKL dgemm completed ==\n"
" == at %.5f milliseconds using %d thread(s) ==\n\n", (s_elapsed * 1000),
i);
}
```

Παρατηρήστε τα αποτελέσματα που λάβατε και απαντήστε στα εξής:

(A13) Ο χρόνος εκτέλεσης αυξάνεται ή μειώνεται όσο ανεβαίνει ο αριθμός των νημάτων;

(A14) Τι γίνεται στην περίπτωση που θέσουμε παραπάνω από το μέγιστο αριθμό νημάτων προς εκτέλεση;