جَامعة
المَنارة
MANARA UNIVERSITY

البرمجة التفرعية

Parallel Programming

Dr.-Eng. Samer Sulaiman

2023-2024

مفردات المنهاج

- أساسيات البرمجة التفرعية
  - مقدمة
  - معامل التسريع
  - أنواع الأنظمة المتعددة المعالجات والبرمجيات الداعمة لها
  - موازنة الأعباء وتحمل الخلل
  - تطبيقات البرمجة التفرعية
  - أشكال معالجة المعطيات على التوازي
- الحواسيب التفرعية
  - تصنيف فلاين Flynn's Classification Scheme
  - شبكات الربط الداخلية Interconnection Networks
- مبادئ تصميم الخوارزميات المتوازية
  - مفاهيم أساسية
  - الإجرائيات والمقابلة
  - تقنيات التقسيم
- البرمجيات الداعمة للبرمجة التفرعية
  - المعتمدة على الذاكرة المشتركة
  - المعتمدة على تمرير الرسائل
- أدوات البرمجة التفرعية
  - برمجة الذاكرة الموزعة باستخدام MP
  - برمجة الخيوط باستخدام Pthreads
- تحليل الأداء Performance Analysis

البرمجة المتوازية
البرمجة عن طريق Multithreading

- البرمجة متعددة الخيوط Multithreading:
  - المتغيرات الشرطية (Condition Variables):
    - طريقة استخدام المتغيرات الشرطية:
      - إنشاء وتدمير المتغيرات الشرطية
      - طريقة الاستخدام:
        - يجب الاعلان عن المتغيرات الشرطية بالنوع pthread_cond_t،
        - يجب تهيئتها قبل استخدامها. وهنالك طريقتين للتهيئة المتغيرات هما:
          - ساكنة (Statically)، عند الاعلان عنها، مثال:
        - pthread_cond_t myconvar = PTHREAD_COND_INITIALIZER;
        - ديناميكية (Dynamically) باستخدام الاجرائية ()pthread_cond_init
          - قيمة تعريف المتغير الشرطي (ID) الذي ينشأ ترجع للخيط الذي يستدعيها عبر المدخل (condition parameter)
          - يمكن إعداد صفات كائن المتغير الفرعي . attr
        - الاجرائية ()pthread_condattr_init والاجرائية ()pthread_condattr_destroy يستخدمان لإنشاء وتدمير كائنات الصفات للمتغيرات الفرعية (condition variable attribute objects)
        - الاجرائية ()pthread_cond_destroy تستخدم لتحرير المتغير الشرطي الذي لا نحتاج إليه.

البرمجة المتوازية
البرمجة عن طريق Multithreading

- البرمجة متعددة الخيوط Multithreading:
  - المتغيرات الشرطية (Condition Variables):
    - طريقة استخدام المتغيرات الشرطية:
      - الانتظار والتأشير في المتغيرات الفرعية:

- pthread_cond_wait (condition,mutex)
- pthread_cond_signal (condition)
- pthread_cond_broadcast (condition)

- طريقة الاستخدام:
  - الاجرائية pthread_cond_wait() تحجز الخيط المستدعي (calling thread) حتى يتحقق الشرط المحدد (condition is signaled)
    - يجب استدعاء هذه الاجرائية عندما يكون mutex مغلق،
    - يحرر mutex تلقائيا بينما هو منتظر.
    - بعد استلام الاشارة واستيقاظ الخيط، سيغلق mutex تلقائيا ويستخدم بواسطة الخيط.
    - سيكون المبرمج مسؤولا عن فتح mutex عندما ينتهي الخيط منه.

البرمجة المتوازية
البرمجة عن طريق Multithreading

- البرمجة متعددة الخيوط Multithreading:
  - المتغيرات الشرطية (Condition Variables):
    - طريقة استخدام المتغيرات الشرطية:
      - الانتظار والتأشير في المتغيرات الفرعية:
        - طريقة الاستخدام:
          - الاجرائية ()pthread_cond_signal تستخدم لإرسال إشارة إلى (إيقاظ) خيط آخر (الخيط الذي ينتظر المتغير الشرطي).
            - يجب أن يستدعى بعد غلق mutex، ويجب أن يفتح mutex لتكتمل الإجرائية ()pthread_cond_wait عملها
          - الاجرائية ()pthread_cond_broadcast تستخدم بدلا من ()pthread_cond_signal إذا كان هنالك أكثر من خيط محجوز في حالة الانتظار(blocking wait state)
          - استدعاء الاجرائية ()pthread_cond_signal قبل الروتين ()pthread_cond_wait يعتبر خطأ منطقي logical error
        - ملاحظة
          - غلق وفتح المتغير mutex بطريقة سليمة ضروي عند استخدام هذه الاجرائيات.
          - مثال:
            - فشل غلق الـ mutex قبل استدعاء ()pthread_cond_wait قد يمنعه من الحجز.
            - الفشل في فتح (unlock) الـ mutex بعد استدعاء ()pthread_cond_signal قد لا يسمح للاجرائية ()pthread_cond_wait التي تعمل معها بالاكتمال (تبقى محجوزة)

البرمجة المتوازية
البرمجة عن طريق Multithreading

- البرمجة متعددة الخيوط Multithreading:
  - المتغيرات الشرطية (Condition Variables):
    - مثال:

```
#include <pthread.h>
#include <iostream>
#define NUM_THREADS 3
#define TCOUNT 10
#define COUNT_LIMIT 12
int count = 0;
pthread_mutex_t count_mutex;
pthread_cond_t count_threshold_cv;
void *inc_count(void *t) {
int i;
long my_id = (long)t;
for (i=0; i < TCOUNT; i++) {
pthread_mutex_lock(&count_mutex);
count++;
if (count == COUNT_LIMIT) {
printf("inc_count(): thread %ld,
count = %d Threshold reached. ",my_id,
count);
pthread_cond_signal(&count_threshold_cv);
printf("Just sent signal.\n");
}
printf("inc_count(): thread %ld, count = %d, unlocking
mutex\n",
my_id, count);
pthread_mutex_unlock(&count_mutex); }
pthread_exit(NULL);
return NULL; }
void *watch_count(void *t) {
long my_id = (long)t;
printf("Starting watch_count(): thread %ld\n", my_id);
pthread_mutex_lock(&count_mutex);
while (count < COUNT_LIMIT) {
printf("watch_count(): thread %ld Count= %d. Going
into wait...\n", my_id,count);
pthread_cond_wait(&count_threshold_cv,
&count_mutex);
printf("watch_count(): thread %ld Condition signal
received. Count= %d\n", my_id,count);
printf("watch_count(): thread %ld Updating the value of
count...\n", my_id,count);
count += 125;
printf("watch_count(): thread %ld count now = %d.\n",
my_id, count);
}
printf("watch_count(): thread %ld Unlocking mutex.\n",
my_id);
pthread_mutex_unlock(&count_mutex);
pthread_exit(NULL);
return NULL;
}
```

البرمجة المتوازية
البرمجة عن طريق Multithreading

- البرمجة متعددة الخيوط Multithreading:
  - المتغيرات الشرطية (Condition Variables):
    - مثال:

```
int main(int argc, char *argv[]) {
int i, rc;
long t1=1, t2=2, t3=3;
pthread_t threads[3];
pthread_attr_t attr;
pthread_mutex_init(&count_mutex, NULL);
pthread_cond_init (&count_threshold_cv, NULL);
pthread_attr_init(&attr);
pthread_attr_setdetachstate(&attr,
PTHREAD_CREATE_JOINABLE);
pthread_create(&threads[0], &attr,
watch_count, (void *)t1);
pthread_create(&threads[1], &attr,
inc_count, (void *)t2);
pthread_create(&threads[2], &attr,
inc_count, (void *)t3);
```

```
for (i = 0; i < NUM_THREADS; i++) {
pthread_join(threads[i], NULL);
}
printf ("Main(): Waited and joined with %d threads. Final value of count = %d. Done.\n",
NUM_THREADS, count);
pthread_attr_destroy(&attr);
pthread_mutex_destroy(&count_mutex);
pthread_cond_destroy(&count_threshold_cv);
system("pause");
pthread_exit (NULL);
}
```