I488111

(19)中華民國智慧財產局

(12)發明說明書公告本 **(11)證書號數：TW I488111 B**

(45)公告日： 中華民國 104 (2015) 年 06 月 11 日

(21)申請案號：102140063 (22)申請日： 中華民國 102 (2013) 年 11 月 05 日

(51)Int. Cl.： *G06F9/42 (2006.01)*

(30)優先權：2012/11/05 美國 61/722,661

2012/12/21 美國 13/724,202

(71)申請人：輝達公司 (美國) NVIDIA CORPORATION (US)

美國

(72)發明人：林元 LIN, YUAN (US)；查奎巴迪 卡烏丹 CHAKRABARTI, GAUTAM (IN)；瑪拉賽 傑帝普 MARATHE, JAYDEEP (IN)；權五官 KWON, OKWAN (KR)；賽伯尼 阿米特 SABNE, AMIT (IN)

(74)代理人：李宗德

(56)參考文獻：

US 20050125774A1 US 20090031290A1

US 20110022672A1 US 20120131309A1

審查人員：何偉權

申請專利範圍項數：10 項 圖式數：4 共 26 頁

(54)名稱

轉譯程式函數以正確處理本地範圍變數的系統及方法和應用其之計算系統

SYSTEM AND METHOD FOR TRANSLATING PROGRAM FUNCTIONS FOR CORRECT HANDLING OF LOCAL-SCOPE VARIABLES AND COMPUTING SYSTEM INCORPORATING THE SAME

(57)摘要

用以轉譯一程式之函數的系統及方法。在一具體實施例中，系統包含：(1)一本地範圍變數識別符，可操作以識別在函數之至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數，以及(2)一函數轉譯器，關聯於本地範圍變數識別符且可操作以轉譯函數之至少某些部份，使得執行緒共享記憶體用以儲存執行緒共享本地範圍變數且執行緒私用記憶體用以儲存執行緒私用本地範圍變數。

A system and method of translating functions of a program. In one embodiment, the system includes: (1) a local-scope variable identifier operable to identify local-scope variables employed in the at least some of the functions as being either thread-shared local-scope variables or thread-private local-scope variables and (2) a function translator associated with the local-scope variable identifier and operable to translate the at least some of the functions to cause thread-shared memory to be employed to store the thread-shared local-scope variables and thread-private memory to be employed to store the thread-private local-scope variables.

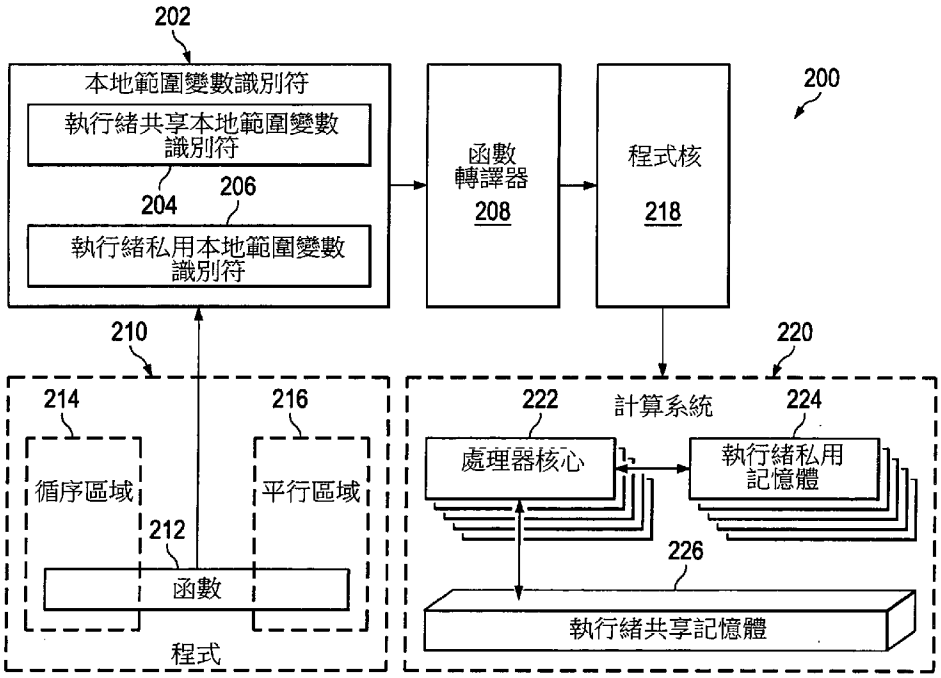

圖2

200・・・系統

202・・・本地範圍變數識別符

204・・・執行緒共享本地範圍變數識別符

206・・・執行緒私用本地範圍變數識別符

208・・・函數轉譯器

210・・・程式

212・・・函數

214・・・循序區域

216・・・平行區域

218・・・程式核

220・・・計算系統

222・・・處理器核心

224・・・執行緒私用記憶體

226・・・執行緒共享記憶體

發明摘要

公告本

※ 申請案號：102140063

※ 申請日：102 年 11 月 5 日　　※IPC 分類：G06F9/42 (2006.01)

【發明名稱】

轉譯程式函數以正確處理本地範圍變數的系統及方法和應用其之計算系統

SYSTEM AND METHOD FOR TRANSLATING PROGRAM FUNCTIONS FOR CORRECT HANDLING OF LOCAL-SCOPE VARIABLES AND COMPUTING SYSTEM INCORPORATING THE SAME

【中文】

用以轉譯一程式之函數的系統及方法。在一具體實施例中，系統包含：(1) 一本地範圍變數識別符，可操作以識別在函數之至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數，以及(2) 一函數轉譯器，關聯於本地範圍變數識別符且可操作以轉譯函數之至少某些部份，使得執行緒共享記憶體用以儲存執行緒共享本地範圍變數且執行緒私用記憶體用以儲存執行緒私用本地範圍變數。

【英文】

A system and method of translating functions of a program. In one embodiment, the system includes: (1) a local-scope variable identifier operable to identify local-scope variables employed in the at least some of the functions as being either thread-shared local-scope variables or thread-private local-scope variables and (2) a function translator associated with the local-scope variable identifier and operable to translate the at least some of the functions to cause

thread-shared memory to be employed to store the thread-shared local-scope variables and thread-private memory to be employed to store the thread-private local-scope variables.

【代表圖】

【本案指定代表圖】：圖 2。

【本代表圖之符號簡單說明】：

200　系統

202　本地範圍變數識別符

204　執行緒共享本地範圍變數識別符

206　執行緒私用本地範圍變數識別符

208　函數轉譯器

210　程式

212　函數

214　循序區域

216　平行區域

218　程式核

220　計算系統

222　處理器核心

224　執行緒私用記憶體

226　執行緒共享記憶體

【本案若有化學式時，請揭示最能顯示發明特徵的化學式】：無。

發明專利說明書

（本說明書格式、順序，請勿任意更動）

【發明名稱】

轉譯程式函數以正確處理本地範圍變數的系統及方法和應用其之計算系統

SYSTEM AND METHOD FOR TRANSLATING PROGRAM FUNCTIONS FOR CORRECT HANDLING OF LOCAL-SCOPE VARIABLES AND COMPUTING SYSTEM INCORPORATING THE SAME

【相關申請案】

【0001】 本申請案係主張美國專利臨時申請案案號61/722,661之優先權，其由Lin等人於2012年11月5日提出申請，標題為「使用一群組的執行緒執行循序碼(EXECUTING SEQUENTIAL CODE USING A GROUP OF THREADS)」，以及主張美國專利申請案案號13/724,202之優先權，其由Lin等人於2012年12月21日提出申請，標題為「轉譯程式函數以正確處理本地範圍變數的系統及方法和應用其之計算系統(SYSTEM AND METHOD FOR TRANSLATING PROGRAM FUNCTIONS FOR CORRECT HANDLING OF LOCAL-SCOPE VARIABLES AND COMPUTING SYSTEM INCORPORATING THE SAME)」，兩案皆與本申請案共同讓與且併入本文作為參考。

【技術領域】

【0002】 本發明一般係有關計算系統，特別是關於用以轉譯程式碼函數以正確處理本地範圍變數的系統及方法和應用其之計算系統。

【先前技術】

【0003】 如熟此技藝者所熟知，軟體程式利用變數，為此需分配儲存空間於記憶體中。儲存空間通常以堆疊的形式分配。

【0004】 某些程式將其變數組織為層級以簡化其資料結構。舉例來說，某些變數(稱作本地範圍變數)可僅用於程式的一函數中(函數範圍)或一敘述區塊中(區塊範圍)。其他變數(稱作通用範圍變數)可由全部程式所使用。舉例來說，以下的表格1提出描述本地範圍變數之兩層級的偽碼：陣列a[]為函數範圍變數，而陣列b[]為區塊範圍變數。

```
void foo() {
int a[100];
   if (...)
      int b[100];
      ...
   }
}
```

表格 1-函數範圍及區塊範圍變數的範例

【0005】 為了理解表格1及本發明揭露內容的其餘部份，術語「foo」及「bar」為函數的任意名稱。因此，任何函數可代替「foo」或「bar」。

【0006】 某些程式為平行程式，其能夠在平行處理器中平行地執行，例如圖形處理單元(GPU)。平行程式具有無法在執行緒中平行執行之程式碼的循序區域以及可以在執行緒中平行執行之程式碼的平行區域。在平行程式中，多個執行緒需要獲得對某些本地範圍變數(其由程式化模組根據所開發的程式而定義)的存取。OpenMP及OpenACC為用以開發平行程式之傳統程式化模組的兩個範例。以下的表格2及3提出分別描述OpenMP及OpenACC範例的偽碼。

```
void foo() {
  int a[100];
  #pragma omp parallel shared(a)
  {
    // a[] can be accessed by all threads in the OpenMP region.
  }
}
```

表格 2-OpenMP 範例

```
void foo() {
    int a[100];
    #pragma acc loop worker
    {
        // a[] can be accessed by all workers in the gang
    }
}
```

表格 3-OpenACC 範例

【發明內容】

【0007】 一態樣提供一種用以轉譯一程式之函數的系統。在一具體實施例中，系統包含：(1) 一本地範圍變數識別符，可操作以識別在函數之至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數，以及(2) 一函數轉譯器，關聯於本地範圍變數識別符且可操作以轉譯函數之至少某些部份，使得執行緒共享記憶體用以儲存執行緒共享本地範圍變數且執行緒私用記憶體用以儲存執行緒私用本地範圍變數

【0008】 另一態樣提供一種用以轉譯一程式之函數的方法。在一具體實施例中，方法包含：(1) 識別函數之至少某些部份為在程式的循序區域期間執行或在程式的平行區域期間執行，(2) 識別在函數之至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍

變數，以及(3) 分 配執行緒私用記憶體用於在程式之平行區域期間執行之函數中所使用之執行緒私用本地範圍變數及執行緒共享本地範圍變數的儲存，以及分配執行緒共享記憶體用於在程式之循序區域期間執行之函數中所使用之執行緒共享本地範圍變數的儲存。

【0009】 在另一具體實施例中，方法包含：(1) 產生函數之至少某些部份之對應的共享複製及私用複製，以及(2) 在程式之循序區域的執行過程中引用共享複製以代替對應函數以及在程式之平行區域的執行過程中引用私用複製以代替對應函數。

【圖式簡單說明】

【0010】 現在將參照以下描述並連同所附隨圖式，其中：

【0011】 圖1為能夠平行處理且可操作以包含或實現用以轉譯函數以正確處理本地範圍變數之系統及方法的一計算系統的方塊圖；

【0012】 圖2為用以轉譯函數以正確處理本地範圍變數之系統的一具體實施例的方塊圖；

【0013】 圖3為用以轉譯函數以正確處理本地範圍變數之方法的一具體實施例的流程圖；以及

【0014】 圖4為用以轉譯函數以正確處理本地範圍變數之方法的另一具體實施例的流程圖。

【實施方式】

【0015】 某些計算架構(包含以通用中央處理單元(CPUs)為中心者)

一般提供一記憶體結構，其係預設為由全部的給定程式所共享，使得程式中的所有區塊、及區塊中的所有函數具有對所有變數的存取。在此架構中分配儲存空間很簡單；其係分配在共享記憶體結構中。

【0016】 然而，某些計算系統(如GPUs)使用階層型的記憶體架構，其中某些記憶體為執行緒私用(僅可由一給定的執行緒存取)，而其他記憶體為執行緒共享(可由在例如程式的一給定區塊或其整體中的多個執行緒存取)。

【0017】 將理解到，在某些程式中所定義的本地範圍變數係專用於一執行緒中，而其他的則需在執行緒之間共享。然而，更將理解到，在許多程式化模型(例如OpenMP或OpenACC)中的程式並不區別執行緒私用及執行緒共享本地範圍變數。

【0018】 一個保守但簡單的解決方法可能為針對所有的本地範圍變數而分配執行緒共享記憶體中的儲存空間。然而本文中將理解到，執行緒共享記憶體通常太小而無法包含所有的本地範圍變數。另一方法可能為針對所有的本地範圍變數而分配執行緒私用記憶體中的儲存空間。然而，本地範圍變數接著將無法以簡單的方式在執行緒之間共享。相反地，本地範圍變數將必須在執行緒私用記憶體之間複製以實現共享，其係冗長且將減慢程式執行。另一方法可能為重寫程式本體，以將所呼叫函數併入對其呼叫的函數中，此技術稱作插入(inlining)。然而，這將模糊化程式的結構，使其難以理解及修改。另一方法可能為手動地分析程式，手動地指定本地範圍變數儲存於執行緒私用或執行緒共享記憶體中。然而，此方法費力且容易出錯。

【0019】 本文中將理解到，需要自動機制用以轉譯程式的函數，以識別需要執行緒共享記憶體的本地範圍變數(執行緒共享本地範圍變數)。一旦識別，可接著正確地處理本地範圍變數。更特別地，執行緒共享記憶體中的儲存空間可針對執行緒共享本地範圍變數而分配，而執行緒本地記憶體中的儲存空間可針對不需要共享記憶體的本地範圍變數(執行緒私用本地範圍變數)而分配。

【0020】 因此，本文介紹用以轉譯函數以正確處理在執行緒共享及私用前後文中之本地範圍變數之系統及方法的各種具體實施例。一般而言，系統及方法具體實施例識別本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數，並基於其識別及使用其之函數所被執行的區域(循序區域或平行區域)而分配執行緒私用或執行緒共享記憶體用以儲存本地範圍變數。

【0021】 在更詳細地描述新穎系統及方法之各種具體實施例之前，現在將描述代表性的計算系統。

【0022】 圖1為可實施本發明一或多個態樣於其中之計算系統100的方塊圖。計算系統100包含多個執行緒處理器或核心106，其組織為執行緒群組104或「執行包(warps)」。計算系統100包含J個執行緒群組104-1到104-J，其每一者具有K個核心106-1到106-K。在某些具體實施例中，執行緒群組104-1到104-J可更組織為一或多個執行緒區塊102。在一特定具體實施例中，每一執行緒群組104具有三十二個核心106。其他具體實施例可包含少至四個核心於一執行緒群組中以及多達數萬個。某些具體實施例將核心106組織為單一執行緒群組104，而其他具體實施例可具有數百甚至數千個

執行緒群組104。計算系統100的其他具體實施例可僅將核心106組織為執行緒群組104，省略了執行緒區塊組織等級。

【0023】 計算系統100更包含管線控制單元108、共享記憶體110、及關聯於執行緒群組104-1到104-J的本地記憶體112-1到112-J之陣列。管線控制單元108經由資料匯流排114將任務分配給不同的執行緒群組104-1到104-J。管線控制單元108產生、管理、排程、執行、並提供一機制以同步化執行緒群組104-1到104-J。

【0024】 一執行緒群組內的核心106係彼此平行的執行。執行緒群組104-1到104-J經由記憶體匯流排116與共享記憶體110通訊。執行緒群組104-1到104-J經由本地匯流排118-1到118-J分別與本地記憶體112-1到112-J通訊。舉例來說，執行緒群組104-J藉由經本地匯流排118-J通訊而利用本地記憶體112-J。計算系統100的某些具體實施例分配共享記憶體110的共享部份給每一執行緒區塊102並允許執行緒區塊102內的所有執行緒群組104對共享記憶體110的共享部份進行存取。某些具體實施例包含僅使用本地記憶體112的執行緒群組104。許多其他具體實施例包含平衡本地記憶體112及共享記憶體110之使用的執行緒群組104。

【0025】 圖1的具體實施例包含主執行緒群組104-1。剩餘執行緒群組104-2到104-J的每一者係視為「背景工作(worker)」執行緒群組。主執行緒群組104-1包含許多核心，其中一者為主核心106-1，其基本上執行主執行緒。在計算系統100上執行的程式係建構為一序列的核(kernels)。一般而言，每一核在下一核開始前完成執行。在某些具體實施例中，計算系統100可平行地執行多個核，其取決於核的尺寸。每一核係組織為在要核心106上執行

之執行緒的一階層。

【0026】 已描述了代表性的計算系統，現在將更詳細地描述用以轉譯函數以正確處理在執行緒共享及私用前後文中之本地範圍變數的新穎系統及方法的各種具體實施例。

【0027】 圖2為用以轉譯函數以正確處理本地範圍變數之系統200的一具體實施例的方塊圖。系統200包含本地範圍變數識別符202及函數轉譯器208。程式210的函數212由本地範圍變數識別符202及函數轉譯器208所處理以產生可在計算系統220上執行的程式核218。計算系統220包含處理器核心222、執行緒私用記憶體224及執行緒共享記憶體226。所有處理器核心222可獲得對執行緒共享記憶體226的存取。每一處理器核心222可獲得其對執行緒私用記憶體224之相關區塊的存取。

【0028】 程式210可分配為循序區域214及平行區域216。程式210內的函數212可從循序區域214及平行區域216的任一者執行，且通常在程式210之序列中不同點從兩者執行。函數212使用有關函數212之一特定函數的通用範圍或本地範圍變數。特定函數內的本地範圍變數有執行緒共享及執行緒私用變數。執行緒共享本地範圍變數可由程式核218內的所有平行執行緒所存取且在計算系統220上執行。執行緒私用本地範圍變數僅可由將變數實例化的執行緒所存取。

【0029】 本地範圍變數識別符202包含執行緒共享本地範圍變數識別符204及執行緒私用本地範圍變數識別符206。執行緒共享本地範圍變數識別符204藉由判定以下而在函數212的單一函數上操作：(1)單一函數是否含有平行結構，其中平行結構內的不同執行緒可具有對本地範圍變數的存

取，或(2)單一函數內的本地範圍變數是否脫離至執行的其他執行緒(在脫離分析下)。若這些條件的任一者為真，本地範圍變數為一執行緒共享本地範圍變數。否則，執行緒私用本地範圍變數識別符206係識別本地範圍變數為執行緒私用本地範圍。

【0030】 函數轉譯器208藉由在程式核218中產生有關應如何針對在單一函數中的每一本地範圍變數分配記憶體之指令而在函數212的單一函數上操作。針對執行緒共享本地範圍變數，分配執行緒共享記憶體226的區塊。針對執行緒私用本地範圍變數，分配執行緒私用記憶體224的區塊。當程式核218在計算系統220內的處理器核心222上執行時，即在執行時間，針對變數而分配記憶體的程序將根據由函數轉譯器208所產生的指令而開始。

【0031】 本文實現用以轉譯一函數的兩種方法，使得本地範圍變數選擇性地分配在執行緒共享記憶體或執行緒私用記憶體。本文所實現的方法並不需要「插入(inlining)」所有函數呼叫或整體程式分析。本文所實現的方法為通用且支援個別的編譯與鏈接，其為建立大型模組軟體的基礎。

【0032】 本文所述的兩個方法利用平行程式化模型(如OpenMP及OpenACC)的兩個特性。第一為不是所有本地範圍變數都需要被共享。本文所實現的方法承認此事實且識別必要的執行緒共享本地範圍變數。第二個特性為當一函數在程式的平行區域內被呼叫，執行緒無法獲得對其他執行緒之本地範圍變數的存取。本文所實現的方法係認定當函數內實例化在程式的平行區域內被呼叫，本地範圍變數不需被共享。

【0033】 本文中所揭露之兩方法中的第一者的一特定具體實施例包含以下程序：

1. 執行緒共享記憶體的區塊係針對一共享堆疊而預先分配。執行緒私用記憶體的區塊係針對關聯於每一執行緒的私用堆疊而預先分配。

2. 函數內的執行緒共享本地範圍變數及執行緒私用本地範圍變數係識別如下：

a. 若函數包含平行結構且平行結構內的不同執行緒可根據程式化模型而具有對本地範圍變數的存取，則本地範圍變數係識別為一執行緒共享本地範圍變數。

b. 若本地範圍變數脫離，則變數係識別為一執行緒共享本地範圍變數。

c. 所有其他的本地範圍變數係識別為執行緒私用本地範圍變數。

3. 函數 foo()係複製為兩個額外的函數：共享複製 foo_shared()及私用複製 foo_private()。

a. 在 foo_shared()中，

i. 在程序 2 中所識別的執行緒共享本地範圍變數係分配於執行緒共享記憶體中。在程序 2 中所識別的執行緒私用本地範圍變數係分配於執行緒私用記憶體中。

ii. 若 foo()含有對另一函數 bar()的直接呼叫且 bar()在平行結構內被呼叫，則對 bar()的呼叫係轉譯為對在 foo_shared()中之 bar_private()的呼叫。若 bar()在平行結

構外被呼叫，則對 bar()的呼叫係轉譯為對在 foo_shared()中之 bar_shared()的呼叫。

iii. 若 foo()不含有任何平行區域，則對另一函數 bar()的直接呼叫係轉譯為對在 foo_shared()中之 bar_shared()的呼叫。

b. 在 foo_private()中，

i. 所有本地範圍變數(不論其在程序 2 中被識別為什麼)係分配於執行緒私用記憶體中。

ii. 對另一函數 bar()的呼叫係轉譯為對其私用複製 bar_private()的呼叫，因為私用複製將僅從平行區域內被呼叫。

4. 原始函數 foo()係轉譯為以下形式：

```
return_type foo(arguments) {
        if (in_parallel_flag == false)
            return foo_shared(arguments);
        else
            return foo_private(arguments);
    }
```

表格 4-轉譯的函數

其中「in_parallel_flag」為以下程序 6 中所描述的每一執行緒旗標。當定址函數，將使用轉譯形式的位址。

5. 在程序 3 及 4 之後，只有私用複製直接在任何平行區域內呼叫。只有共享複製直接在任何平行區域外呼叫。對來自程序 4 的轉譯版本做出間接呼叫。只有執行緒共享或可能的執行緒共享本地範圍變被放入執行緒共享記憶體中。

6. 每一執行緒旗標係用以標示執行緒的狀態。旗標的數值為「真」或「假」。當執行緒進入平行區域，旗標設定為「真」。當執行緒離開平行區域，旗標設定為「假」。

【0034】 圖3為用以轉譯函數以正確處理本地範圍變數之方法的一具體實施例的流程圖。方法開始於開始步驟310。在步驟320中，執行緒共享記憶體及執行緒私用記憶體係針對本地範圍變數而預先分配。在一具體實施例中，執行緒共享記憶體為區塊範圍記憶體。

【0035】 在步驟330中，產生函數之至少某些部份的對應共享複製及私用複製。在步驟340中，在函數中所使用的本地範圍變數係識別為執行緒共享本地範圍變數或執行緒私用本地範圍變數。在步驟350中，針對共享複製之其中至少一者，分配執行緒共享記憶體用於執行緒共享本地範圍變數的儲存以及分配執行緒私用記憶體用於執行緒私用本地範圍變數的儲存。在步驟360中，針對私用複製之其中至少一者，分配執行緒私用記憶體用於所有本地範圍變數的儲存。在判斷步驟370中，判定程式的平行區域是否被執行。若是，在步驟380中，在執行過程中引用私用複製以代替對應的函數。若否，在步驟390中，引用共享複製以代替對應的函數。

【0036】 在一具體實施例中，步驟380、390的引用係藉由以下而實現：(1)在共享複製的至少一者中，(1a)將在平行結構中所做出之函數的呼叫轉譯為函數之私用複製的呼叫，以及(1b)將在循序結構中所做出之函數的呼叫轉譯為函數之共享複製的呼叫，以及(2)在私用複製的至少一者中，將函數的呼叫轉譯為函數之私用複製的呼叫。在一相關的具體實施例中，步驟

380、390的引用係藉由使用在執行程式中之函數的至少某些部份的位址而實現。在另一相關的具體實施例中，步驟380、390的引用係藉由轉譯函數的至少某些部份以使引用被執行而實現。在一相關的具體實施例中，分配包含使用一旗標，其在循序區域的執行過程中具有第一狀態且在平行區域的執行過程中具有第二狀態。方法結束於結束步驟395。

【0037】 本文中所揭露之兩方法中的第二者的一特定具體實施例包含以下程序：

1. 執行緒共享記憶體的區塊係針對一共享堆疊而預先分配。執行緒私用記憶體的區塊係針對關聯於每一執行緒的私用堆疊而預先分配。

2. 函數內的執行緒共享本地範圍變數及執行緒私用本地範圍變數係識別如下：

a. 若函數包含平行結構且平行結構內的不同執行緒可根據程式化模型而具有對本地範圍變數的存取，則本地範圍變數係識別為一執行緒共享本地範圍變數。

b. 若本地範圍變數脫離，則變數係識別為一執行緒共享本地範圍變數。

c. 所有其他的本地範圍變數係識別為執行緒私用本地範圍變數。

3. 在一函數 foo()中，在程序 2 中所識別的所有執行緒私用本地範圍變數係分配於執行緒私用記憶體中。在程序 2 中所識別

的執行緒共享本地範圍變數的每一者係基於函數是否在串行或平行前後文內被呼叫而有條件地分配。

```
return_type foo(arguments) {
 if (in _parallel_flag == false)
    alloca te thread-shared local-scope variables on shared stack;
  else
    allocate thread-shared local-scope variables on private stack;
    allocate private local-scope variables on private stack;
    <function-body>
}
```

表格 5-轉譯的函數

4. 每一執行緒旗標係用以標示執行緒的狀態。旗標的數值為「真」或「假」。當執行緒進入平行區域，旗標設定為「真」。當執行緒離開平行區域，旗標設定為「假」。

【0038】 圖4為用以轉譯函數以正確處理本地範圍變數之方法的另一具體實施例的流程圖。方法開始於開始步驟410。在步驟420中，執行緒共享記憶體及執行緒私用記憶體係針對本地範圍變數的儲存而預先分配。在一具體實施例中，執行緒共享記憶體為區塊範圍記憶體。

【0039】 在步驟430中，函數之至少某些部份係識別為在程式的循序區域期間執行或在程式的平行區域期間執行。在步驟440中，在函數之至少某些部份中所使用的本地範圍變數係識別為執行緒共享本地範圍變數或執行緒私用本地範圍變數。在一具體實施例中，識別本地範圍變數包含：若一函數包含一平行結構且多個執行緒可根據一程式化模型獲得對平行結構內之本地範圍變數的存取或本地範圍變數脫離函數時，識別一本地範圍變

數為一執行緒共享本地範圍變數，且識別剩餘的本地範圍變數為執行緒私用本地範圍變數。

【0040】 在步驟450中，執行緒私用記憶體係分配為用於在程式之平行區域期間執行之函數中所使用之執行緒私用本地範圍變數及執行緒共享本地範圍變數的儲存。在步驟460中，執行緒共享記憶體係分配為用於在程式之循序區域期間執行之函數中所使用之執行緒共享本地範圍變數的儲存。在一具體實施例中，轉譯函數的至少某些部份，以使分配被執行。在一相關的具體實施例中，分配包含使用一旗標，其在循序區域的執行過程中具有第一狀態且在平行區域的執行過程中具有第二狀態。方法結束於結束步驟470。

【0041】 熟習本申請案相關技藝者將理解到，可對所述具體實施例做出其他及更多添加、刪減、替換或修改。

【符號說明】

100 計算系統

102 執行緒區塊

104-1~104-J 執行緒群組

106-1~106-K 核心

108 管線控制單元

110 共享記憶體

112-1~112-J 本地記憶體

114 資料匯流排

116 記憶體匯流排

118-J 本地匯流排

200 系統

202 本地範圍變數識別符

204 執行緒共享本地範圍變數識別符

206 執行緒私用本地範圍變數識別符

208 函數轉譯器

210 程式

212 函數

214 循序區域

216 平行區域

218 程式核

220 計算系統

222 處理器核心

224 執行緒私用記憶體

226 執行緒共享記憶體

【生物材料寄存】

國內寄存資訊【請依寄存機構、日期、號碼順序註記】

國外寄存資訊【請依寄存國家、機構、日期、號碼順序註記】

【序列表】(請換頁單獨記載)

申請專利範圍

1. 一種用以轉譯一程式之函數的系統，包含：

一本地範圍變數識別符，可操作以識別在該函數之至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數；以及

一函數轉譯器，關聯於該本地範圍變數識別符且可操作以轉譯該函數之該至少某些部份，使得執行緒共享記憶體用以儲存該執行緒共享本地範圍變數且執行緒私用記憶體用以儲存該執行緒私用本地範圍變數。

2. 如申請專利範圍第1項所述之系統，其中該本地範圍變數識別符包含：

一執行緒共享本地範圍變數識別符，可操作以在若一函數包含一平行結構且多個執行緒可根據一程式化模型獲得對該平行結構內之該本地範圍變數的存取或該本地範圍變數脫離該函數時，識別一本地範圍變數為一執行緒共享本地範圍變數；以及

一執行緒私用本地範圍變數識別符，關聯於該執行緒共享本地範圍變數識別符且可操作以識別剩餘的本地範圍變數為執行緒私用本地範圍變數。

3. 如申請專利範圍第2項所述之系統，其中該程式化模型係選自由以下所組成之群組：

一OpenMP程式化模型；以及

—OpenACC程式化模型。

4. 如申請專利範圍第1項所述之系統，其中該函數轉譯器更可操作以使用一旗標，該旗標在該程式之循序區域的執行過程中具有一第一狀態且在該程式之平行區域的執行過程中具有一第二狀態。

5. 如申請專利範圍第1項所述之系統，其中該執行緒共享記憶體為區塊範圍記憶體。

6. 如申請專利範圍第1項所述之系統，更包含組態以預先分配該執行緒共享記憶體及執行緒私用記憶體的一記憶體預先分配器。

7. 一種用以轉譯一程式之函數的方法，包含：

產生該函數之至少某些部份的對應共享複製及私用複製；以及

在該程式之循序區域的執行過程中引用該共享複製以代替該對應函數以及在該程式之平行區域的執行過程中引用該私用複製以代替該對應函數。

8. 如申請專利範圍第7項所述之方法，更包含：

識別在該函數中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數；

針對該共享複製之其中至少一者，分配執行緒共享記憶體用於該執行緒共享本地範圍變數的儲存以及分配執行緒私用記憶體用於該執行緒私用本地範圍變數的儲存；以及

針對該私用複製之其中至少一者，分配執行緒私用記憶體用於所有本地範圍變數的儲存。

9. 一種用以轉譯一程式之函數的方法，包含：

識別該函數之至少某些部份為在該程式的循序區域期間執行或在該程式的平行區域期間執行；

識別在該函數之該至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數；以及

分配執行緒私用記憶體用於在該程式之該平行區域期間執行之函數中所使用之該執行緒私用本地範圍變數及執行緒共享本地範圍變數的儲存，以及分配執行緒共享記憶體用於在該程式之該循序區域期間執行之函數中所使用之執行緒共享本地範圍變數的儲存。

10. 如申請專利範圍第9項所述之方法，其中識別該本地範圍變數之該步驟包含：

若一函數包含一平行結構且多個執行緒可根據一程式化模型獲得對該平行結構內之該本地範圍變數的存取或該本地範圍變數脫離該函數時，識別一本地範圍變數為一執行緒共享本地範圍變數；以及

識別剩餘的本地範圍變數為執行緒私用本地範圍變數。

圖式

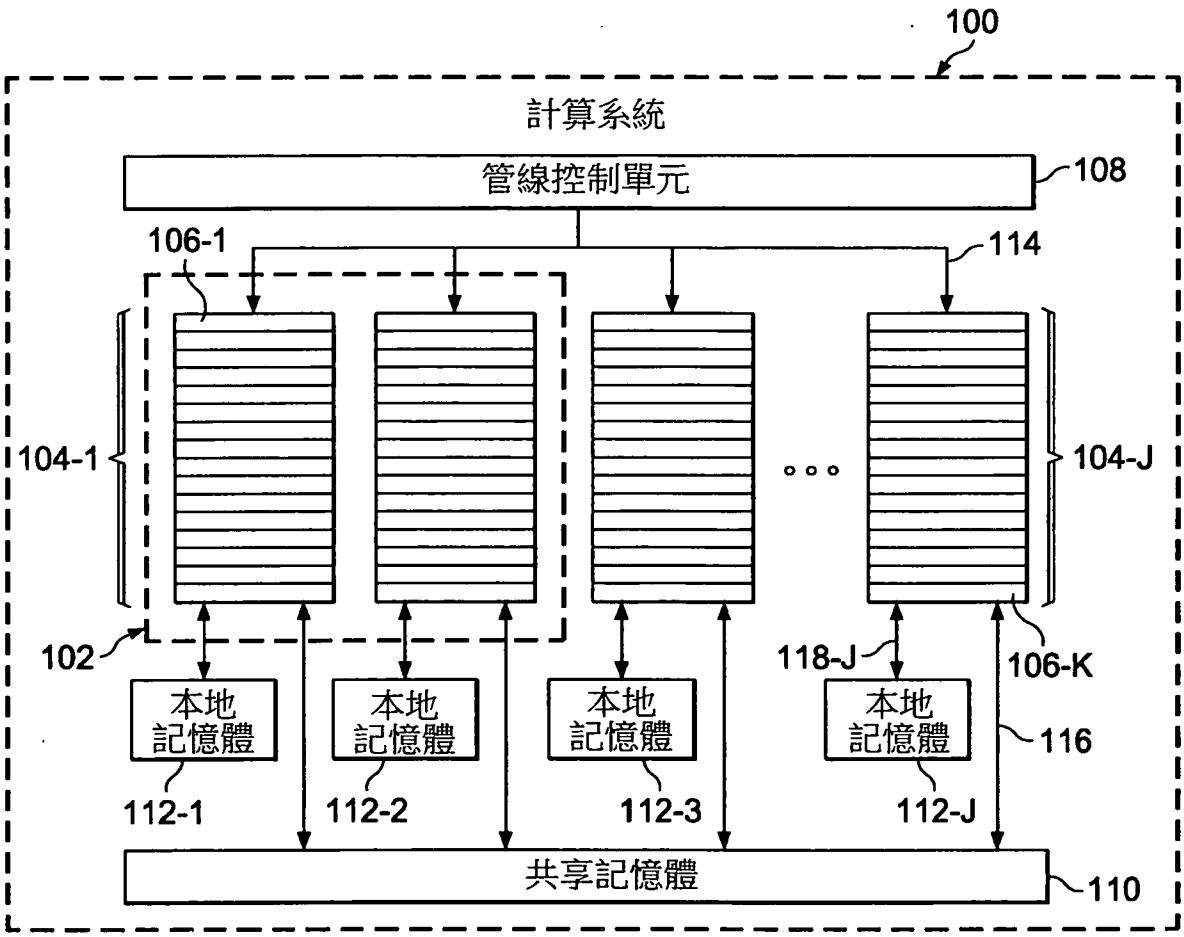
100
計算系統
管線控制單元
108
106-1
114
104-1
104-J
102
118-J
106-K
本地記憶體
本地記憶體
本地記憶體
本地記憶體
116
112-1
112-2
112-3
112-J
共享記憶體
110

圖1

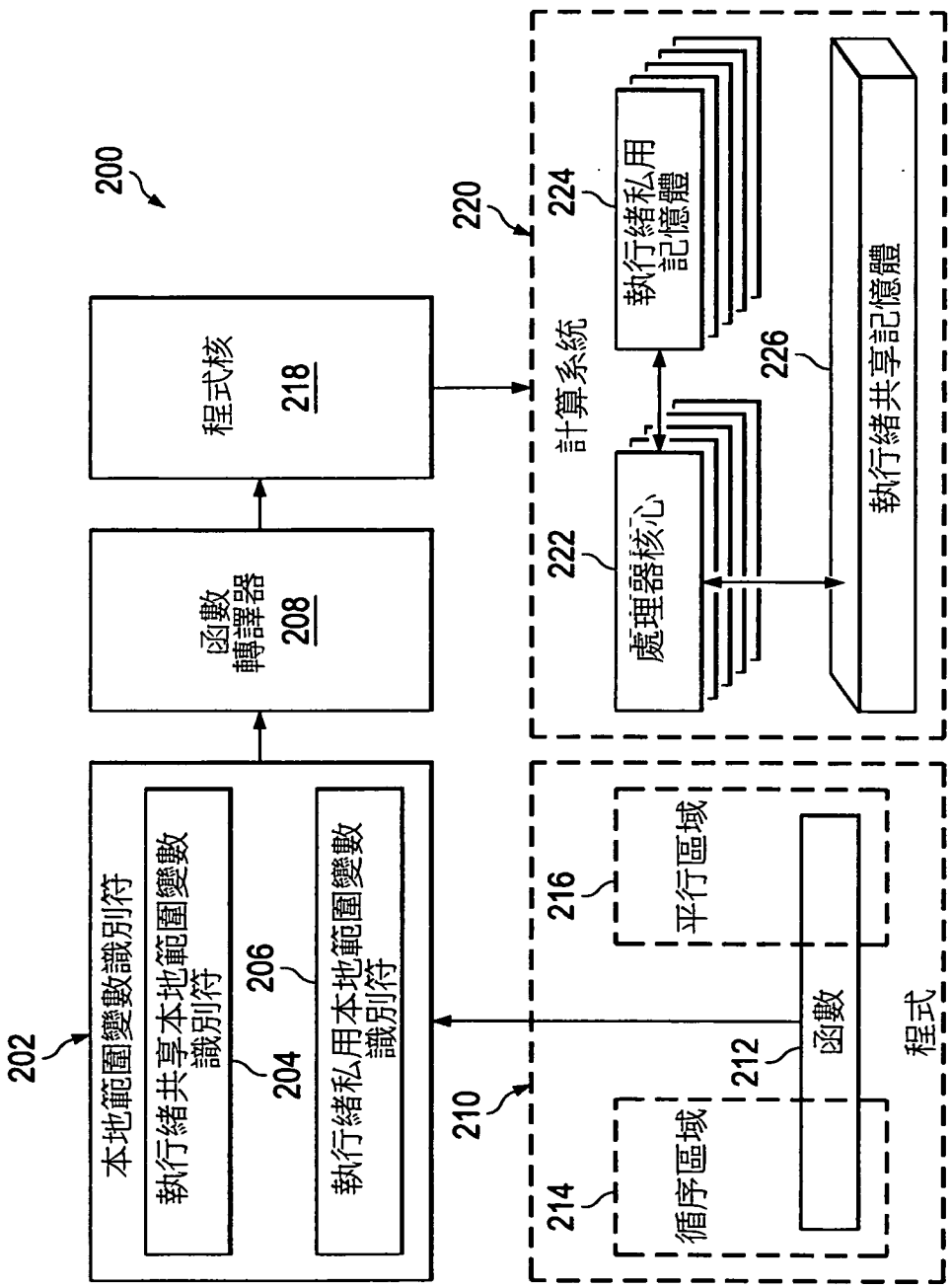
200
202
本地範圍變數識別符
執行緒共享本地範圍變數識別符
204
206
執行緒私用本地範圍變數識別符
函數轉譯器 208
程式核 218
210
220
計算系統
222
處理器核心
224
執行緒私用記憶體
226
執行緒共享記憶體
程式
212
函數
214
循序區域
216
平行區域

圖2

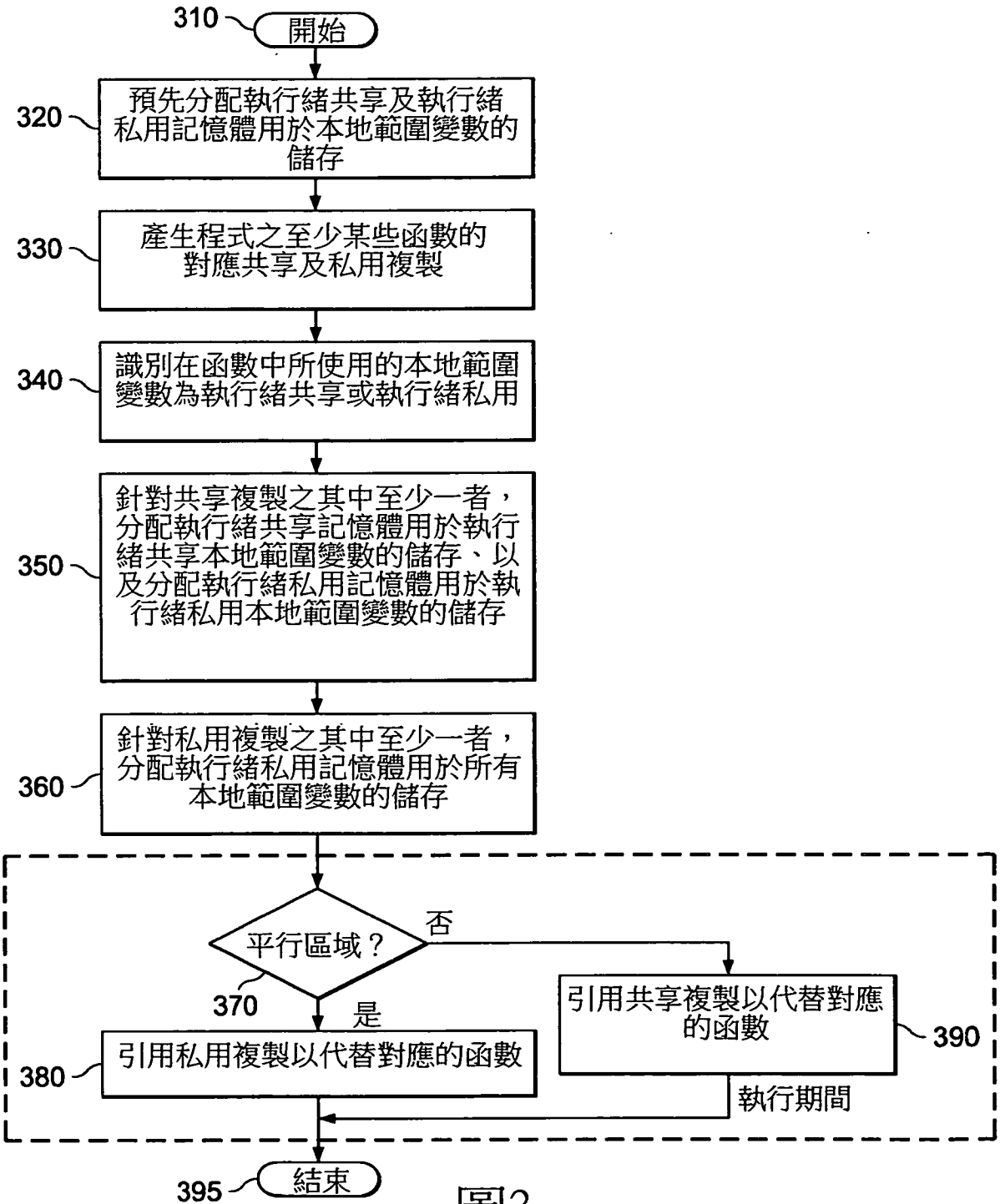
310
開始
320
預先分配執行緒共享及執行緒私用記憶體用於本地範圍變數的儲存
330
產生程式之至少某些函數的對應共享及私用複製
340
識別在函數中所使用的本地範圍變數為執行緒共享或執行緒私用
350
針對共享複製之其中至少一者，分配執行緒共享記憶體用於執行緒共享本地範圍變數的儲存、以及分配執行緒私用記憶體用於執行緒私用本地範圍變數的儲存
360
針對私用複製之其中至少一者，分配執行緒私用記憶體用於所有本地範圍變數的儲存
平行區域？
370
否
是
引用共享複製以代替對應的函數
390
引用私用複製以代替對應的函數
380
執行期間
395
結束

圖3

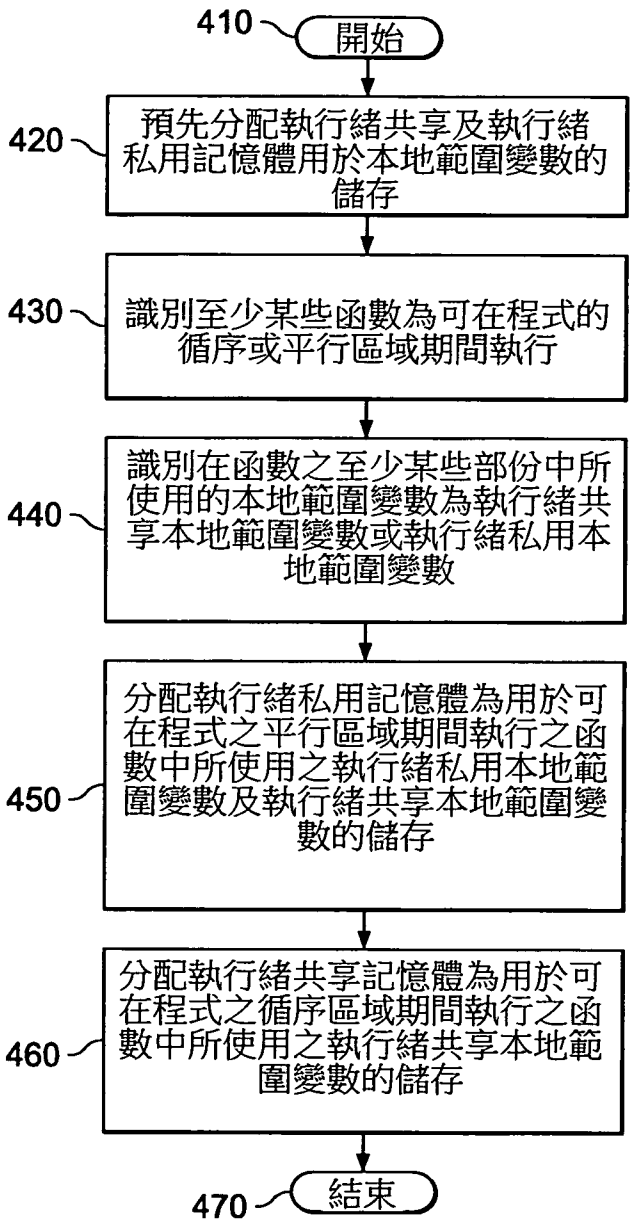
410
開始
420
預先分配執行緒共享及執行緒私用記憶體用於本地範圍變數的儲存
430
識別至少某些函數為可在程式的循序或平行區域期間執行
440
識別在函數之至少某些部份中所使用的本地範圍變數為執行緒共享本地範圍變數或執行緒私用本地範圍變數
450
分配執行緒私用記憶體為用於可在程式之平行區域期間執行之函數中所使用之執行緒私用本地範圍變數及執行緒共享本地範圍變數的儲存
460
分配執行緒共享記憶體為用於可在程式之循序區域期間執行之函數中所使用之執行緒共享本地範圍變數的儲存
470
結束

圖4