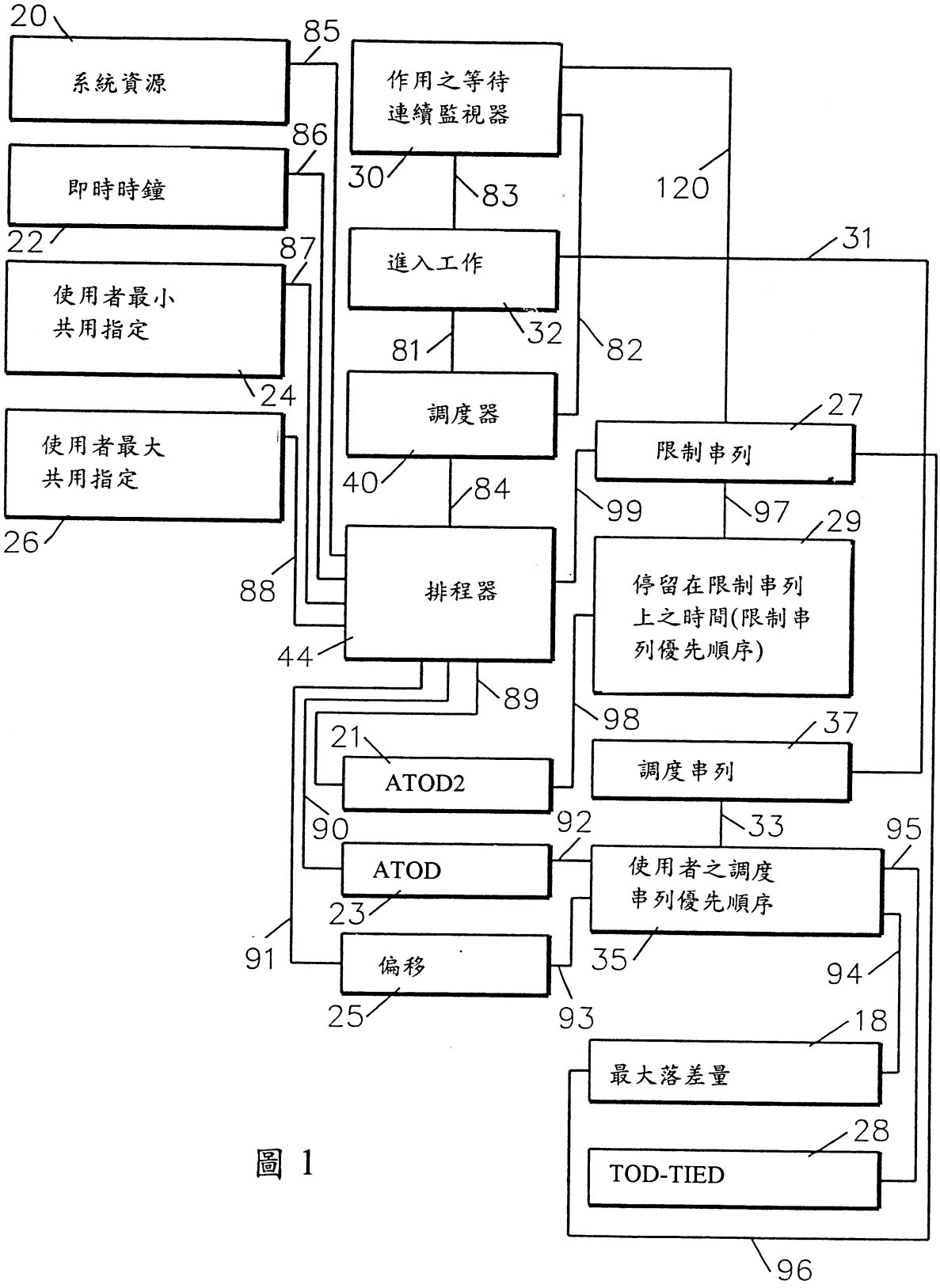
20
系統資源
85
作用之等待
連續監視器
86
即時時鐘
30
83
120
22
87
使用者最小
共用指定
進入工作
31
81
32
82
24
使用者最大
共用指定
調度器
27
限制串列
40
84
99
97
26
29
88
排程器
停留在限制串列
上之時間(限制串
列優先順序)
44
89
98
21
37
ATOD2
調度串列
90
92
33
95
ATOD
使用者之調度
串列優先順序
23
91
偏移
35
94
25
93
18
最大落差量
28
TOD-TIED
96

圖 1

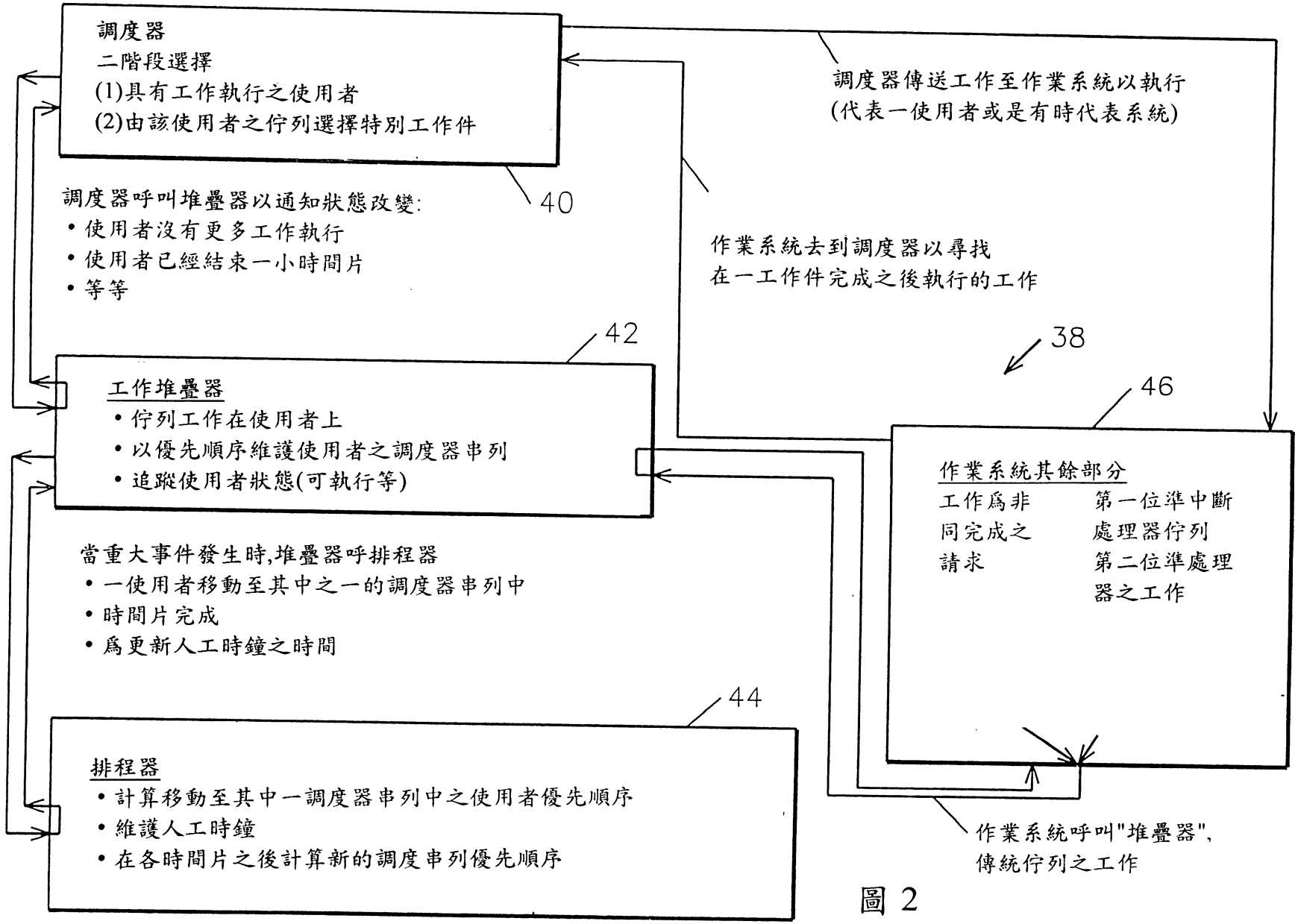
調度器
二階段選擇
(1)具有工作執行之使用者
(2)由該使用者之佇列選擇特別工作件
40
調度器傳送工作至作業系統以執行
(代表一使用者或是有時代表系統)
調度器呼叫堆疊器以通知狀態改變:
• 使用者沒有更多工作執行
• 使用者已經結束一小時間片
• 等等
作業系統去到調度器以尋找
在一工作件完成之後執行的工作
38
42
46
工作堆疊器
• 佇列工作在使用者上
• 以優先順序維護使用者之調度器串列
• 追蹤使用者狀態(可執行等)
作業系統其餘部分
工作爲非同完成之請求
第一位準中斷處理器佇列
第二位準處理器之工作
當重大事件發生時,堆疊器呼排程器
• 一使用者移動至其中之一的調度器串列中
• 時間片完成
• 爲更新人工時鐘之時間
44
排程器
• 計算移動至其中一調度器串列中之使用者優先順序
• 維護人工時鐘
• 在各時間片之後計算新的調度串列優先順序
作業系統呼叫"堆疊器",
傳統佇列之工作

圖 2

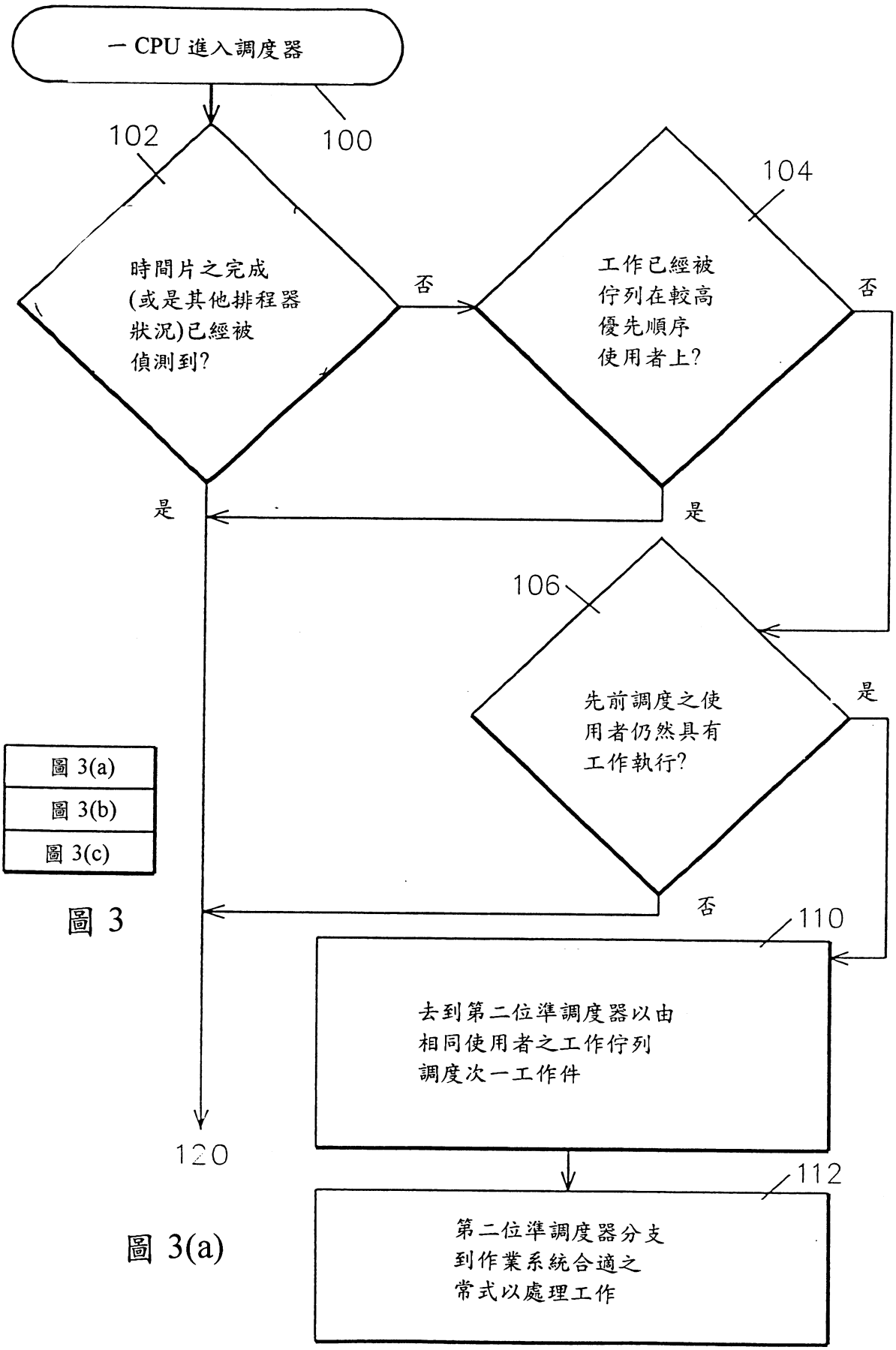

圖 3(a)
圖 3(b)
圖 3(c)

圖 3

圖 3(a)

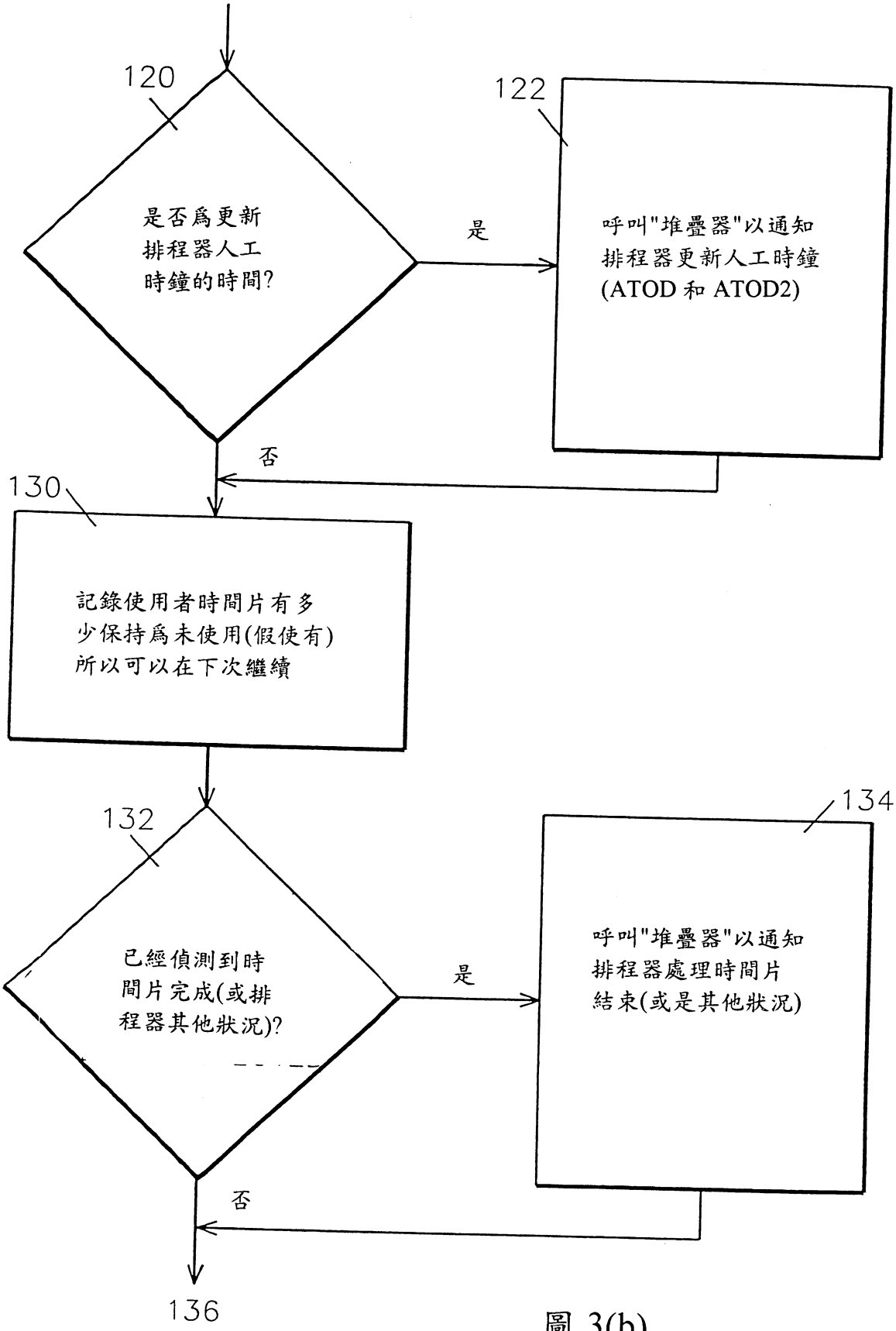
120
是否爲更新
排程器人工
時鐘的時間?
是
122
呼叫"堆疊器"以通知
排程器更新人工時鐘
(ATOD 和 ATOD2)
否
130
記錄使用者時間片有多
少保持爲未使用(假使有)
所以可以在下次繼續
132
已經偵測到時
間片完成(或排
程器其他狀況)?
是
134
呼叫"堆疊器"以通知
排程器處理時間片
結束(或是其他狀況)
否
136

圖 3(b)

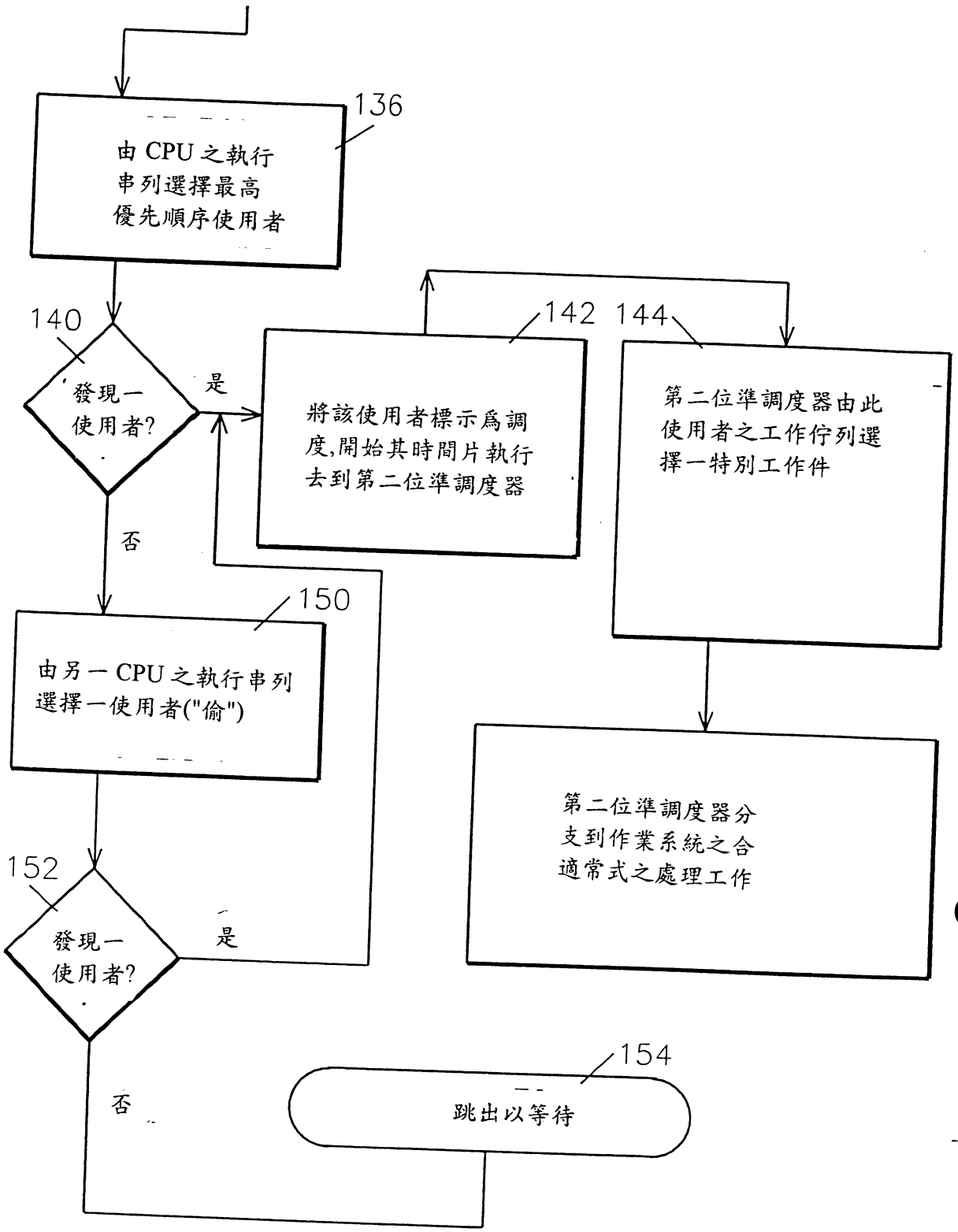
136
由 CPU 之執行
串列選擇最高
優先順序使用者
140
發現一
使用者?
是
否
142
將該使用者標示爲調
度,開始其時間片執行
去到第二位準調度器
144
第二位準調度器由此
使用者之工作佇列選
擇一特別工作件
150
由另一 CPU 之執行串列
選擇一使用者("偷")
152
發現一
使用者?
是
否
第二位準調度器分
支到作業系統之合
適常式之處理工作
154
跳出以等待

圖 3(c)

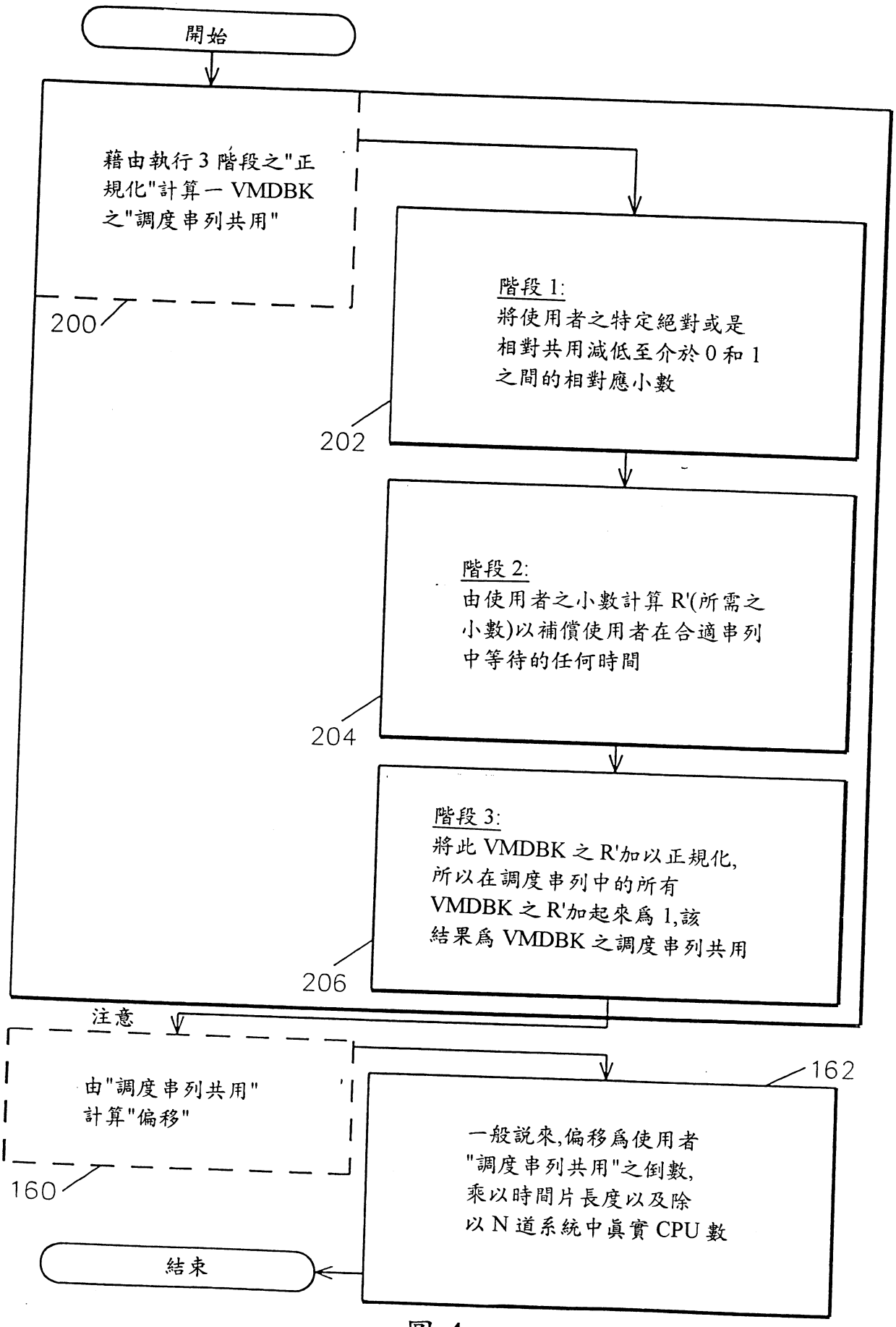
開始
藉由執行 3 階段之"正規化"計算一 VMDBK 之"調度串列共用"
200
階段 1:
將使用者之特定絕對或是相對共用減低至介於 0 和 1 之間的相對應小數
202
階段 2:
由使用者之小數計算 R'(所需之小數)以補償使用者在合適串列中等待的任何時間
204
階段 3:
將此 VMDBK 之 R'加以正規化,所以在調度串列中的所有 VMDBK 之 R'加起來爲 1,該結果爲 VMDBK 之調度串列共用
206
注意
由"調度串列共用"計算"偏移"
160
一般說來,偏移爲使用者"調度串列共用"之倒數,乘以時間片長度以及除以 N 道系統中眞實 CPU 數
162
結束

圖 4

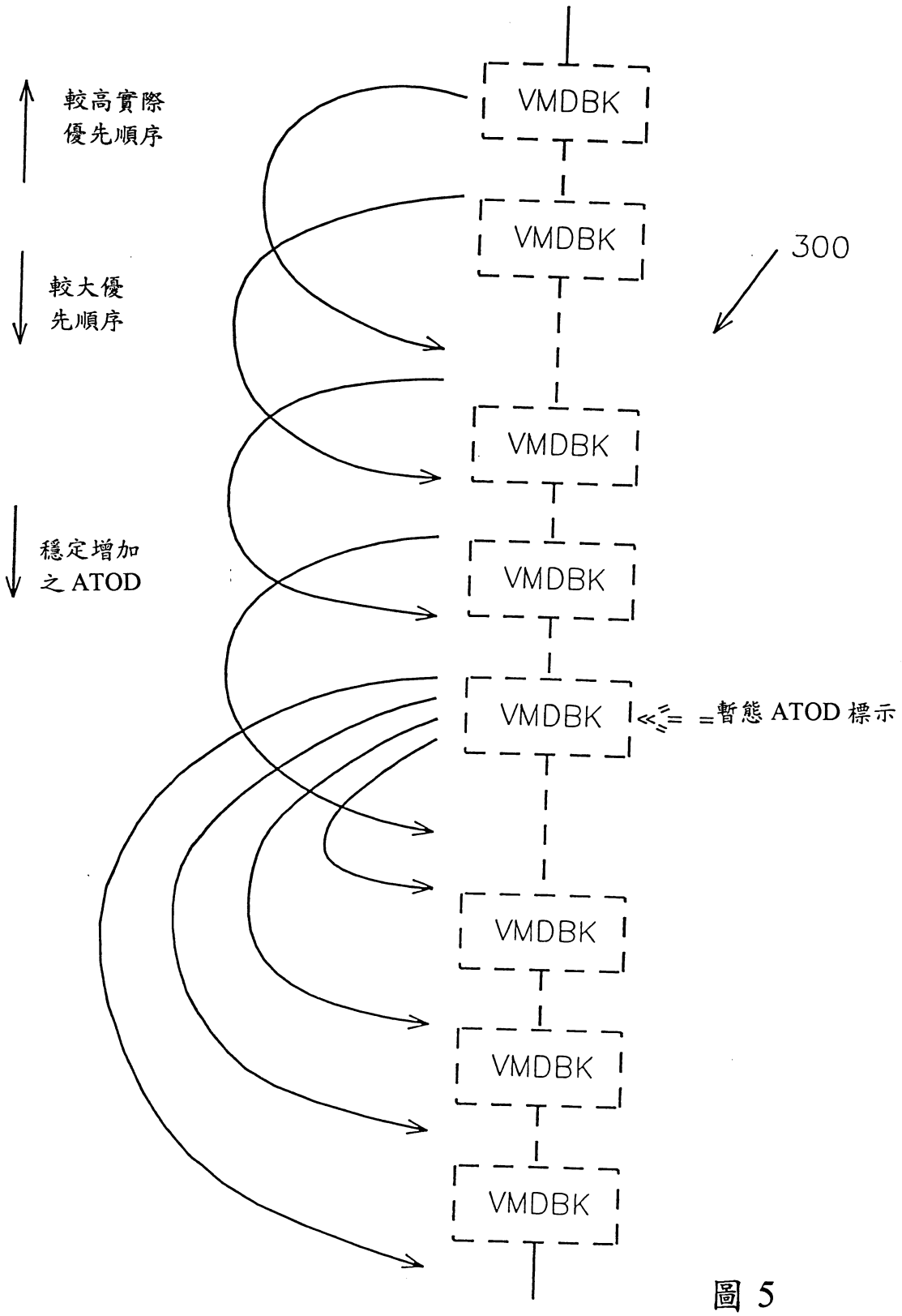
較高實際
優先順序
較大優
先順序
穩定增加
之 ATOD
VMDBK
VMDBK
300
VMDBK
VMDBK
VMDBK
=暫態 ATOD 標示
VMDBK
VMDBK
VMDBK

圖 5

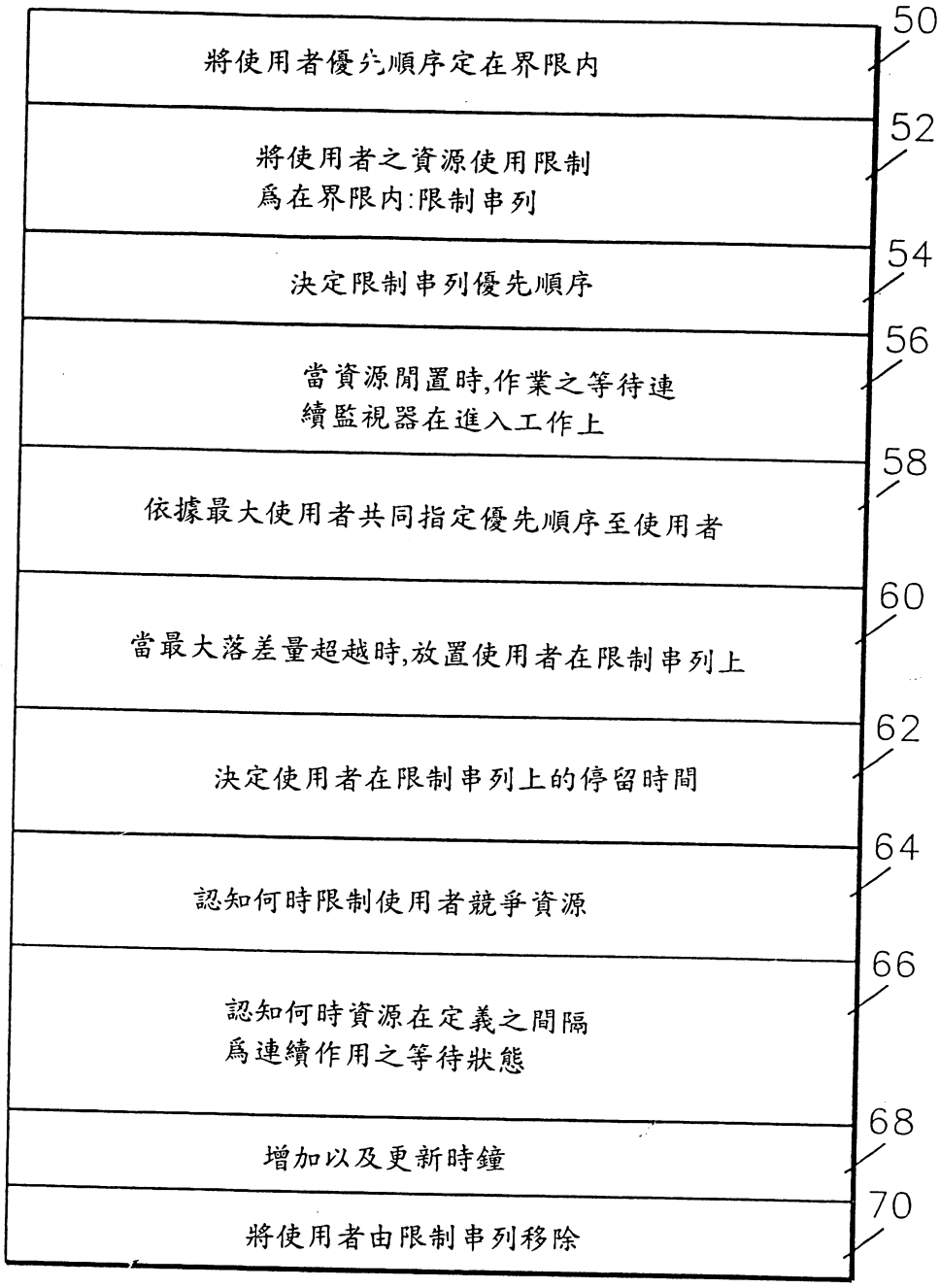
將使用者優先順序定在界限內
50
將使用者之資源使用限制
爲在界限內:限制串列
52
決定限制串列優先順序
54
當資源閒置時,作業之等待連
續監視器在進入工作上
56
依據最大使用者共同指定優先順序至使用者
58
當最大落差量超越時,放置使用者在限制串列上
60
決定使用者在限制串列上的停留時間
62
認知何時限制使用者競爭資源
64
認知何時資源在定義之間隔
爲連續作用之等待狀態
66
增加以及更新時鐘
68
將使用者由限制串列移除
70

圖 6

91年6月4日修正補充

公告本

申請日期	88、12、23
案　　號	088122782
類　　別	G06F 9/46

A4
C4

中文說明書修正本 (91年6月)

（以上各欄由本局填註）

發明 新型 專利說明書 526452

一、發明 新型名稱	中　文	用以將系統資源排程之系統及方法
	英　文	SYSTEM AND METHOD FOR SCHEDULING SYSTEM RESOURCES
二、發明 創作人	姓　名	1.約翰 F. 哈利斯 2.馬可 J. 羅倫克
	國　籍	均美國
	住、居所	1.美國紐約州恩得威爾市耶爾街2606號 2.美國紐約州恩得威爾市佛那路528號
三、申請人	姓　名（名稱）	美商萬國商業機器公司
	國　籍	美國
	住、居所（事務所）	美國紐約州阿蒙市新果園路
	代表人 姓　名	傑拉德 羅森賽

本紙張尺度適用中國國家標準(CNS) A4規格(210×297公釐)

頒請委員明示：91年6月4日所提之修正本有無變更原實質內容？是否准予修正。

裝 訂 線

91.

A6
B6

（由本局填寫）

承辦人代碼：
大　　類：
ＩＰＣ分類：

本案已向：

國（地區）　申請專利，申請日期：　　案號：　　，□有　□無主張優先權

美國　　1999年3月25日　09/276,525　　☑有□無主張優先權

有關微生物已寄存於：　　寄存日期：　　，寄存號碼：

五、發明說明（ 1 ）

發明背景

發明領域

本發明係關於一系統排程器。更特別地係關於將有調度驅動多處理器系統資格的資源排程至使用者。

背景技藝

排程器係提供用以在作業系統中以控制由使用者消耗的系統資源量。由G. A. Davison於6/2/94申請之尚在審理中的美國專利申請案案號為第S/N 08/252,864號之申請案說明此種系統。

往往在電腦系統中，許多程式或是使用者需要並行執行。可能有至少一可使用的處理器執行多程式。任一處理器只可以在任一時間執行一程式或是異動。假使至少二程式在相同時間需要執行，而且只有一處理器或是只有少於需要執行之程式數目的處理器時，作業系統必須決定何程式可以在任一時間使用該處理器或是各處理器。通常此決定為依據各程式之相對優先順序。通常，亦且在等待狀態之程式如那些等待一慢速I/O操作完成之程式，將被"懸置"以及所以不論優先順序為何將沒有資格爭奪處理器。

除決定何程式可以在一特別時間利用該處理器或是各處理器以外，作業系統亦必須決定所選擇之程式可以在開始執行之後利用該個別處理器多長時間，所以為相等(或是可能較低)優先順序之程式亦可以輪流執行。具有多數已知技術用於決定各程式在開始執行之後必須被許可利用該個別處理器多長時間。在最簡單的技術中，最高優先順序之程

五、發明說明（ 2 ）

式維持處理器之使用，直到該程式本身完成使用該處理器為止(或是懸置)。然而，此技術可能允許此程式"霸佔"該處理器，以及使其他程式"挨餓"。在另一種已知技術中，所選擇用於執行之各程式為給予總體處理器時間之共用或是總體處理器時間"片"。程式之共用可以為為相等或是較佳地依據各種不同因素而加權。一種已知技術如下列方式將共用加權。其中一些程式被定義為命令總體處理器時間之"絕對"或是明確共用，而其他程式被定義為命令剩餘處理器時間之"相對"共用。該絕對共用"由上層移去"而相對共用以其相對共用依比率劃分剩餘時間。例如，第一程式以40%絕對共用加以定義，第二程式以100相對共用加以定義以及第三程式以300相對共用加以定義。在此方案中，第一程式將命令40%之總體處理器時間，第二程式將命令15%之總體處理器時間而第三程式將命令45%之總體處理器時間。

一程式可能不需要其全部絕對或是相對共用。以此絕對/相對共用環境分配過剩之一種已知技術為將把所有過剩時間給予具有最高絕對或是相對共用之程式。假使該過剩時間為小部分總體處理器時間時，此已知技術將足夠滿足需要。然而，在許多案例中，該過剩時間為大量，如總體處理器時間之一半，所以將所有過剩時間分配給一程式可能不是最佳方式。

使用者有權利使用之系統相對共用為依據目前競爭系統資源的所有相對共用使用者優先順序比率給予資格。然

五、發明說明（ 3 ）

而，所有系統資源不需要為相對共用使用者可利用。假使具有絕對共用使用者競爭資源時，那些絕對共用使用者為在剩餘之系統共用在相對共用使用者之間依比率劃分之前首先給予其部分之系統。這些關係通常表示如下：

剩餘共用＝100％系統－所有絕對共用總體

之後剩餘共用在相對共用使用者之間依比率劃分：

相對使用者之共用＝剩餘共用*(使用者相對共用/所有相對使用者之總體共用)

一小時間片為一既定使用者許可由排程器再次檢視而不具中斷的處理器時間量。小時間片的大小與機器速度以及可能之用戶設定有關。當一使用者被調度時，CPU時間以小時間片的時間量加以設定。當時間量期滿時，此計時器減少以及呈現中斷，因此發出信號至排程器表示使用者的小時間片已經結束。

一受限之使用者，亦即"限制固定"使用者，為限制值定義為代表擁有最大量處理器資源之一使用者。此限制值代表最大使用共用，以及用以計算使用者的調度優先順序。通常，限制串列提供為放置那些超越或是將超越其最大許可共用之處理器資源的限制固定使用者。當一使用者在限制串列上(或是另外由限制狀態加以特徵化)時，資源不會為

五、發明說明（ 4 ）

該使用者調度使用。

一人工日時鐘(ATOD)於調度使用者工作時使用。ATOD值以和機器的日時鐘(TOD)相同速率增加，但是只有在使用者工作(以和非使用者工作如作業系統工作區別)由機器完成期間增加。在目前系統中，至少一部分依賴ATOD排程系統資源，縱使當CPU資源使用時，受限制之共用使用者可能不被允許達到其最大使用共用。明顯地是，在具有閒置處理器資源時間的低負載情況中，人工日時鐘(ATOD)值在閒置期間移動不夠快速。所以，正執行之那些限制固定使用者的優先順序相對於ATOD較其應該之移動更快速以及其過早地進入限制串列(也就是說，設定限制狀態)。甚至於，使用者可能不需要維持在限制串列上，因為只有作用之(非閒置)處理器監視該串列。在低負載情況中(當沒有足夠之工作使所有處理器保持忙碌的時間)，較少作用之處理器為作用，以致限制串列較不常被監視。此情況建立介於使用者必須離開限制串列與處理器偵測使用者必須離開限制串列之間的時間增加。迄今為止，沒有一機構在處理器為作用之等待狀態的時間期間將使用者由限制串列移除。在此情況中，具有當處理器沒有使用者工作將執行的時間以及所以在作用之等待機構中迴路，以及在限制串列上之一使用者保持在串列上直到在另一處理器上可能造成限制串列將檢查之其他事件發生為止。

因此，當一現行系統如在上述Davison申請案中說明的系統具有較少使用者在上面，或是當該系統不為忙碌時，當

五、發明說明（ 5 ）

使用者必須能夠具有最高至硬性限制的使用時，系統排程器給予使用者較使用者硬性限制少的CPU時間。在具有非常少使用者在電腦系統上的案例中，說也奇怪使用者得到較應得之CPU時間明顯為少，最高少於20%。此情形在CPU時間為計費的案例中特別會形成使用者抱怨得不到所付費用的所有CPU時間。相反地，CPU時間可能應計費而不計費。

本發明之一目的為提供一種排程器系統和方法，以加強允許使用者在低負載情況中存取系統資源精密接近或是等於其硬性限制。

本發明之另一目的為提供一種改良之排程器，加強系統資源之最佳化計費以及減低使用者在該資源計費上的不安或是可使用性。

本發明之另一目的為提供一種系統和方法，因此監視資料使用為系統使用的排程演算法之一回饋機構以決定使用者優先順序以及決定接下來何使用者必須被許可加以執行。

本發明之另一目的為提供一種系統和方法，用以藉由不會太快將具有限制存取之使用者置放在限制串列上以及不會太慢將使用者由限制串列移除而將系統資源排程至該使用者。

本發明之另一目的為提供一種系統和方法，用於不會太慢因為使資源能夠排程而排程系統源給具有限制存取之使用者，以及不會太快禁止資源之排程。

五、發明說明（ 6 ）

發明總結

系統資源由使用者藉由維持一第一時鐘、維持一第二時鐘而排程使用；在調度處理期間，假使有工作將執行時，只有當該使用者不在等待狀態時，將系統資源之使用依據調度優先順序排程給該使用者；否則以將第一時鐘往前推進一使用者時間；以及以將第二時鐘往前推進一使用者時間和等待時間；以及在等待處理期間，假使有工作將執行時，呼叫調度處理；否則，計時等待時間；以及假使在等待時間超越一預定值時具有一個為等待狀態的使用者時，以增加第一時鐘一等待時間和呼叫調度處理。

本發明其他態樣和優點將由本發明目前較佳具體實施例的下列詳細說明與附圖結合而變得明顯。

圖式之簡單說明

圖1為製作本發明較佳具體實施例之方法和資料結構的系統圖；

圖2為電腦作業系統流程圖；

圖3(a-c)形成調度器流程圖；

圖4為圖2作業系統內排程器的(小部分)流程圖；

圖5解釋調度串列；以及

圖6為製作本發明方法之較佳具體實施例之程式模組圖示。

實施本發明之最佳模式

如本發明較佳具體實施例，為在尚在審理中的專利申請案案號為第S/N 08/252,864號之申請案中的系統改良，之

五、發明說明（ 7 ）

前，一作業系統藉由不會太快將該具有有限之存取的使用者放置在限制串列中而將系統資源排程至該使用者，以及不會太慢將使用者由該限制串列移除。

因此，提供一種系統和方法用於在一'調度驅動'多處理系統中經由建立在一作用之等待常式的監視器之使用限制一特別使用者之CPU使用為一絕對值。該使用之機構為CPU資源必須限制的使用者串列，所以其消耗不會超越該限制值。該使用者特徵為具有'限制狀態'。該限制串列為監視之作用等待以決定在低負載情況下何時該使用者必須由該串列移除(因此重新儲存為'調度狀態')以及因此在CPU可使用時遞送最大CPU使用至使用者。如本發明較佳具體實施例，此作用之等待常式(亦稱之為處理，或是程序)監視在限制串列上的使用者以決定何時為呼叫調度常式之時間以將使用者由該串列移除，利用監視之效能資料並回饋以動態調整排程常式而增強決定使用者優先順序和決定何使用者必須被允許接著執行(調度)之效應。當ATOD2被計算為在使用者限制串列優先順序之一小時間片之內時，一使用者由該調度常式從該限制串列移除。

使用者工作為依據代表使用者時間片結束之調度優先順序值而加以調度執行。此值為日時鐘格式，以及在下文中更進一步說明。

參考圖1，一系統圖解釋本發明較佳具體實施例之程式(資料和方法)結構。這些程式結構包含系統資源20、即時時鐘22、使用者最小共用指定24、使用者最大共用指定

A7
B7

五、發明說明（ 8 ）

26。作用之等待連續監視器30、進入工作32、調度器40、排程器44、ATOD2 21、ATOD 23、偏移25、限制串列27、在限制串列上停留期間(亦稱之為停留時間)29、使用者優先順序界限35、調度串列37、最大降落量18和TOD-TIED 28。介於製作本發明方法之較佳具體實施例之這些結構之間的鏈接，將在下文中解釋。

再次參考圖1，在操作中如以線88表示，假使有工作將執行時，調度器40尋找進入工作32以及使其依據由排程器44設定之優先順序執行。如以線82表示，假使沒有工作將執行時，調度器40呼叫作用之等待連續監視器30。如以線83表示，作用之等待處理30連續監視系統之進入工作32以及更進一步以線82表示，呼叫調度器40以當工作到達時加以執行該工作。如以線84表示，調度器40呼叫排程器44以當使用者達到小時間片結束時重新計算使用者優先順序。如分別以線85、86、87和88表示，排程器44使用為至調度優先順序計算系統資源20、及時時鐘22、使用者最小共用指定24、使用者最大共用指定26之輸入。如以線89表示，排程器44將ATOD2往前推進一實際使用者使用之CPU時間量加上自上次ATOD2更新時的任何等待時間(沒有工作可使用於執行)，而更新ATOD2 21。如以線90表示，排程器44將ATOD往前推進一自上次ATOD更新起的實際使用者之CPU時間量，以及基本上在本發明特定狀況下亦一往前推進一等待時間，而更新ATOD2 23。該更新之ATOD=先前之ATOD加上自上次ATOD重新計算之使用者CPU時間。

五、發明說明（ 9 ）

如以線91表示，排程器44計算使用者偏移25，該偏移將在優先順序計算中使用。當使用者調度優先順序被計算時，偏移25被用為現行ATOD 23值之位移。如以線92和93表示，使用者調度串列優先順序35(亦稱之為期限優先順序或是預測之完成時間)為依據排程器44之計算以及在界限內依據現行ATOD 23值和使用者偏移25之大小加以設定。如以線31和33表示，使用者調度串列優先順序被用以優先順序置放一使用者工作在調度串列37上，以致於調度器40視其為需要執行之進入工作32。排程器44使用ATOD 23計算使用者調度串列優先順序35以將使用者置放在調度串列37上以及當使用者資源消耗需要藉由將其置放在限制串列27上時使用ATOD2 21計算限制串列27優先順序。如以線95表示，計算之上限稱為TOD-TIED 28。如以線94表示，計算之下限稱為最大降落量18。如以線96表示，調度優先順序35由排程器計算為在最大降落量界限18外部之限制共用使用者加入至限制串列27。如以線97和98表示，在限制串列27上之停留期間27的預測依據ATOD2 21製成。如以線99表示，使用者停留在限制串列27上直到由排程器44移除的時間為止。如以線120表示，在低負載情況中，作用之等待處理30監視限制串列27以決定何時一使用者可能必須被移除。

決定ATOD、TOD-TIED、調度優先順序和偏移以及將資源加以排程至使用者所需之計算將在下文中說明。用於計算最大降落量之等式為：

五、發明說明（ 10 ）

最大降落量=ATOD+4*偏移

用於計算偏移之等式為：

偏移=(小時間片-超限)/(cpu數*共用)

其中超限=在使用者先前小時間片上剩餘時間量，假使使用者已經由較高優先順序佔先。

使用者調度優先順序為計算為在優先順序界限最大降落量和TOD-TIED之界限內的整數值。TOD-TIED機構之更完整解釋提供在下文中。

就整體而言，ATOD 23和ATOD2 21為用以累加在系統上之所有使用者的使用者時間和系統等待時間之資料結構(欄位)。單獨使用者時間及/或等待時間不需要追蹤。這些累加被用於預測將發生之特定事件如和系統作用比較之使用者小時間片的結束。

圖2-圖5，以及下列之說明由Davison之尚在審理中申請案案號為S/N 08/252,864號加以改編，以及組成瞭解本發明不可或缺之基本觀念的基本指導，本發明主要為提供於調整ATOD和ATOD2以包含等待時間之機構的改良，以此方式實質上在不會太快增加使用者至限制串列或是不會太慢將該使用者移除之目的上改良。

圖2解釋一作業系統包括調度器40、堆疊器42、排程器44和其餘功能46。該堆疊器為介於作業系統排程器和調度

五、發明說明（ 11 ）

器以及介於調度器和排程器和其餘功能之間的介面。下文為處理流程之高階說明。在一典型方案中，使用者發出一請求至作業系統"其餘功能"46以執行一些工作。該工作請求傳送至記錄使用者狀態為"備妥"之堆疊器42。縱使當使用者可能具有其工作之不同觀點，該堆疊器將由使用者到達之工作視為至少一"異動"。當一使用者狀態由完全閒置性("閒置")變成準備執行("備妥")狀態時，一個"新的"異動開始。亦具有第三狀態為"懸置"，當一使用者仍然具有工作(異動)進行，但是等待一虛擬I/O請求完成以及所以暫時不準備執行時存在。懸置週期不足以宣告一個新的異動。接著，該堆疊器傳送該請求至此使用者工作佇列。該堆疊器亦呼叫排程器以獲得指示此使用者相對於在調度串列37上之其他使用者之調度優先順序35的資訊。之後，該堆疊器置放此使用者(實際上為使用者VMDBK)在調度串列37中以及回轉至作業系統"其他功能"。該調度優先順序使在調度串列上的使用者順序起效應。當作業系統其他功能完成其現行工作件時，其傳送控制至調度器40以決定將執行之次一使用者和工作件。該調度器之後在調度串列37選擇接著將執行之第一備妥使用者以及在使用者工作佇列中的第一工作件。之後，調度器40傳送控制至作業系統其他功能完以開始執行此工作。調度之工作之後執行預定歷時之時間片，例如1毫秒。假使該工作項目在1毫秒時間片結束之前完成時，該調度器由相同使用者工作佇列中選擇另一工作件加以執行。在此時間片結束時，假使該使用者仍

五、發明說明（ 12 ）

然具有工作執行時，則該調度器再次呼叫排程器44以獲得此使用者次一時間片的新優先順序。此新優先順序使得使用者在調度串列中重新定位，依次決定何時相同使用者之此次時間片將開始。介於相同使用者之此次時間片之間的時間部分決定此使用者處理器時間之共用。

無論其絕對或是相對共用，所有使用者於一致歷時之離散時間片加以調度和執行。在前述之例子中，各時間片為1毫秒。一時間片一般依據處理器速度且選擇於此具體實施例之1毫秒僅為例子，以及可以置換以調諧系統。在各使用者(以VMDBK表示)完成一時間片之後，被中斷以及向下移動至限制串列中(重新排優先順序)。此情形一般允許另一使用者使用此處理器。各使用者之處理器共用由介於分配至使用者的時間片和使用者利用各時間片所需之時間之間的間隔決定；各使用者不需要消耗1連續毫秒中的單一時間片。假使該使用者必須等待一I/O操作時，其暫時懸置以及移去處理器(但是保留在調度串列37中)，以致於等待I/O操作完成之時間不計數此使用者時間片。例如，縱使一I/O操作可能需要多數毫秒完成時，一I/O界限使用者將經常能夠啟始以及在使用完單一時間片之前等待至少10 I/O操作完成。亦且，假使工作為具有較高優先順序之另一使用者到達時，該調度器在使用者時間片結束之前可能由其處理移除正執行之使用者以及將其懸置一週期。該作業系統可以藉由控制介於時間片之間的間隔而控制各使用者之處理器時間的實際共用。使用者介於時間片之間的間隔相當於

五、發明說明（ 13 ）

在調度串列37中使用者和調度串列中其他使用者時時刻刻比較之共用大小的倒數。每次一VMDBK完成一時間片時，只有在調度串列37中之該VMDBK重新排優先順序以及重新定位。在單一時間片一段段使用的案例中，排程器不重新計算使用者介於時間片之間的間隔，直到累加時間等於1毫秒時間片為止。用於決定介於時間片之間微小間隔的技術在下文更詳細說明。

在N-路MP系統中之各CPU具有日時(TOD)鐘22和一CPU計時器。該TOD時鐘包括52位元硬體暫存器，每微秒增加一次。當作業系統被啟始(啟動)時，該TOD時鐘被設定以及在設定之後從不改變。在N-路MP系統中，所有N個TOD時鐘為相互準確同步至微秒。該TOD時鐘被設定為如此時間0為1990年1月1日午夜。以52位元(亦即位元0-51) TOD時鐘計算微秒，該TOD時鐘在重覆之前可以運行超過140年。該TOD時鐘從不停止，以及總是增加且為可靠和正確的時間戳記及即時量測。

該CPU計時器包括另一52位元硬體暫存器，由作業系統38使用以將各時間片加以計時。假使以及當系統操作員停止CPU時，該CPU計時器停止，以及只要操作員重新啟動CPU時，該CPU計時器將再次啟動(然而該TOD時鐘連續運行)。當CPU再次啟動時，該CPU計時器和時間片之前進繼續進行。當各使用者開始其時間片時，開始時間由CPU計時器通知。類似地是，使用者每次變成懸置時，該CPU計時器之時間被通知以決定此使用者再次執行時剩餘多少時

五、發明說明（ 14 ）

間。因此，當CPU停止時該使用者正處理，以及相對於CPU計時器量測，不會遺失其任何時間片資格。

該排程器亦具有稱之為人工日時(ATOD)鐘(亦稱之為第二人工日歷鐘ATOD2，稍後說明)之軟體時鐘。該ATOD時鐘類似於硬體TOD時鐘，除了只有一個(不是各CPU具有一個)以及運行較慢以外。CPU總體可使用處理功率之特定量花費在作業系統管理負擔上，不能定為任何特別使用者之屬性。所以，此總體可使用處理時間量不可使用於劃分為使用者之共用。總體處理之百分比為時時刻刻變化之系統管理負擔，使其困難地知道少於100%多少為可使用於使用者之共用。當作業系統正執行管理負擔功能時，此困難度為藉由定義ATOD時鐘將停止而規避，以及之後在遞送共用至使用者的計算中使用ATOD時間而非TOD時間。在ATOD時鐘中，之後時間停止而管理負擔被執行。因此，由該ATOD時鐘量測之100%時間相當於可使用之處理。遞送共用之計算，為依據ATOD時鐘，假使100%處理為可使用於使用者之共用，所以可以計算。

在作業系統38中，優先順序數較低代表較高之實際優先順序。優先順序數始終為使用者改變以及調度順序為依據介於備妥使用者之間的相對優先順序。當一使用者/VMDBK被置放在限制串列37中時，優先順序數由排程器依據個別絕對或是相對共用加以計算。當為剛進入調度串列之使用者計算時，通常優先順序數較現行ATOD時鐘值稍大(些許毫秒)。(然而，實際上，對起先之些許時間片而

五、發明說明（ 15 ）

言，假使非常短異動時較小之優先順序數被用以提供使用者快速回應。此變換如此短，所以不甚影響使用者之共用)。因為ATOD總是往前進，所以較舊、未完成之工作經常具有較低優先順序數以及所以具有較新的工作為高之優先順序，傾向於位在接近調度串列上方。如下文之更詳細說明，調度器在執行串列上(該串列為調度串列37之複製，排除未備妥或是懸置使用者，因此為具有執行狀態使用者之聚集或是串列)由上往下尋找以將使用者定位而執行。如時間往前進，假如使用者因為較高優先順序備妥工作而沒有得到調度或是假如使用者得到調度但是為I/O限制以及花費許多時間"懸置"而沒有完成其時間片時，則使用者之優先順序相對於(1)更新啟動異動之使用者及(2)不是I/O限制之使用者而增加。此二事件分別為真，因為(1)使用者之優先順序數保持固定直到其時間片完成為止，然而啟動新異動之其他使用者於ATOD增加時逐漸指定較高之優先順序數以及(2)更多CPU限制使用者使用許多時間片以及在各使用者之後往下移動。因此，I/O限制使用者傾向於具有較非I/O限制使用者為高的相對優先順序以及通常，定位於接近調度和執行串列之上方以及當備妥執行，亦即在各I/O操作完成之後，被快速提供CPU。

對各使用者而言，具有絕對或是相對共用24、26(在系統目錄中制訂以及有時由系統操作員改變)。對各使用者而言，亦具有制訂之"軟性限制"、"硬性限制"或是"沒有限制"。如未使用之處理時間變成可使用，假使沒有未使用之處

五、發明說明（ 16 ）

理時間時，所有使用者開始接收和其計算之絕對共用與相對共用成比率的過量處理時間。然而，當任何硬性限制使用者達到其硬性限制時，此沒有更進一步之分配。沒有硬性限制使用者可以始終分配超越硬性限制的處理時間之共用。當任何軟性限制使用者達到其軟性限制時，假使任何其他軟性限制使用者已經達到其軟性限制或是任何其他硬性限制使用者已經達到其硬性限制或是假使具有能夠使用過量處理時間之任何未限制使用者時，此軟性限制使用者沒有更進一步之分配。尚未達到其限制之其他軟性限制和硬性限制使用者繼續接收額外共用，直到他們亦達到其個別限制為止。當所有其他軟性限制使用者已經達到其軟性限制以及所有硬性限制使用者已經達到其硬性限制以及沒有非限制使用者能夠使用過量處理時間時，則具有最大絕對共用或是相對共用(假使沒有未使用之處理時間時計算)之軟性限制使用者接收所有剩餘之未使用處理時間。另外，當所有軟性限制使用者已經達到其軟性限制以及所有硬性限制使用者已經達到其硬性限制以及沒有非限制使用者能夠使用過量處理時間時，則所有軟性限制使用者與假使沒有未使用之處理時間時計算之絕對共用或是相對共用成比率共用剩餘之未使用處理時間。

如下文之更詳細解釋，共用分配技術藉由在使用者完成各時間片之後在調度串列中往下重新定位或是偏移各使用者而加以製作。該偏移量為依據絕對或是相對共用之大小。當在完成各時間片之後藉由從剛完成該時間片之前使

五、發明說明（ 17 ）

用者位置啟動偏移而在調度串列中重新定位使用者時，各使用者I/O限制之相對量納入考慮。因此，假使使用者為非常I/O限制時，在完成其時間片之前將被定位在非常接近調度串列上方，所以縱使在完成時間片之後將被重新定位在調度串列高處以及經歷往下偏移。(當一使用者首先進入調度串列時，該偏移由接近ATOD處開始)。

調度器40如圖3(a-c)所解釋加以製作。在步驟100中，一CPU完成使用者之一件工作之執行且通知調度器。回應時，第一位準調度器決定是否使用者已經完成其現行1毫秒時間片(決定102)。假使未完成，則第一位準調度器決定是否較高優先順序使用者正等待處理器(決定104)。假使不是，則第一位準調度器決定是否剛完成一件工作之使用者具有另一件工作執行，亦即在其工作佇列中的另一件工作(決定106)。假使有，則第一位準調度器呼叫第二位準調度器以由此使用者佇列調度次一件工作在剛完成此相同使用者另一件工作的相同CPU上執行(步驟110)。回應時，第二位準調度器選擇此使用者工作佇列上的第一件工作以及傳送控制至作業系統中的合適常式以處理此特別件工作(步驟112)。

再次參考決定102、104和106，假使決定102為是、決定104為是或是決定106為否，則第一位準調度器前進至決定120以決定是否為更新排程器ATOD時鐘(和ATOD2時鐘)的時間，如下文說明。此決定為依據自時鐘上次更新後是否已經期滿至少一毫秒。假使是，則調度器呼叫堆疊器以通

五、發明說明（ 18 ）

知排程器更新ATOD和ATOD2時鐘(步驟122)。接著，第一位準調度器藉由首先記錄使用者時間片有多少剩餘未使用而"未調度"使用者(步驟130)，假使時間片有剩餘。假使決定102導致步驟130，則此時間片沒有剩餘。然而，假使決定104或是決定106在步驟130之前，則剩餘些許之時間片。接著，第一位準調度器決定是否時間片已經完成(以及一些其他排程器情況如是否使用者必須循環回至合適串列之任何情況被檢查)(決定132)。假使是已經完成，第一位準調度器呼叫堆疊器以通知排程器處理時間片之結束或是其他排程情況(步驟134)。接著，第一位準調度器由剛完成異動執行之CPU的執行串列選擇最高優先順序使用者(步驟136)。假使具有此種使用者(決定140)，則第一位準調度器標示此使用者為調度、將時間片(或是由此剩餘之時間)載入至CPU計時器以將此使用者之時間片加以計時且之後去到第二位準調度器(步驟142)。回應時，第二位準調度器由使用者之工作佇列選擇此使用者之第一件工作以及傳送控制至作業系統部分46中的合適常式以處理此特別件工作(步驟144)。

再次參考決定140，假使沒有發現剛完成異動執行之CPU的任何使用者時，第一位準調度器由另一CPU之執行串列選擇最高優先順序使用者(步驟150)。假使發現該使用者(決定152)時，第一位準調度器前進至步驟142以由另一CPU之執行串列調度此使用者。然而，假使在另一CPU之執行串列沒有發現任何使用者時，則第一位準調度器前進

五、發明說明（ 19 ）

至等待狀態(步驟154)。

圖4解釋一部分排程器44。在由第二位準調度器接收(經由堆疊器)將排程之新使用者識別之後，排程器如下決定"調度串列共用"(步驟200)。在步驟202中，排程器以上述先前技藝VM/ESA作業系統之方式換轉使用者絕對或是相對共用為系統處理時間小數部分，亦即所有絕對共用由上方移出(以及假使總體時間大於99%時依比率減少)且剩餘部分在調度串列上之其他使用者之間依據其相對共用比率加以劃分。各使用者之結果為處理器時間小數部分(介於0和1之間)以及所有使用者值加至1。接著，該排程器決定此使用者之"所需小數"(R')以補償使用者花費在等待一"合適"串列之時間(步驟204)。當一使用者準備執行時被置放在合適串列中，但是不具有足夠之記憶體可使用於執行使用者之工作。該所需小數藉由將步驟202中決定之所需小數乘以(在調度串列之時間+在合適串列之時間)/(在調度串列之時間)而決定。各使用者之所需小數在各使用者停留在合適串列之後重新計算且結果用於在調度串列之次一週期。然而，所有使用者經常具有足夠之記憶體，所以至合適串列"旅程"很稀少且經常，步驟204為步驟202結果乘以1之簡單乘法。接著，排程器將此使用者所需小數R'加以正規化，所以在調度串列之所有使用者所需小數加至1(步驟206)。使用者之正規化所需小數R為"調度串列共用"(或是執行串列共用)以及將在下文更詳細說明。由此使用者之調度串列共用，排程器決定沿調度串列和執行串列之"偏移"。粗略

五、發明說明（ 20 ）

地，使用者之偏移為調度串列共用倒數乘以時間片長度以及除以系統中真實CPU數。偏移越小使用者具有更多存取至CPU，反之亦然。使用者偏移之精確值在下文中更詳細說明。

圖5解釋在時間瞬間點之調度串列300。該調度串列包含準備執行之使用者以及懸置之使用者。如上文所表示，在調度串列300上準備執行之使用者亦列表在執行串列上以減少尋找時間，但是調度串列300之解釋為較佳於解釋目的。該調度串列由上方往下尋找；具有最高優先順序之使用者(以及最低優先順序數)依順序列表在上方。當各使用者完成其時間片，假使具有更多工作將執行時，該排程器由現行位置往下等於此使用者偏移25重新定位在調度串列300上之使用者(以及間接在執行串列上)。因此，二因素使在調度串列300上之使用者分佈起效應。第一因素為偏移25之大小。如上述所表示，各使用者偏移之大小相對於和調度串列上其他使用者共用成反相以及和具有不同共用之其他使用者不同。因此，共用越小，偏移越大。此情形大多數時間傾向於將低共用使用者定位在調度串列300底部。第二因素為使用者完成其時間片所需之真實時間量。"I/O限制"使用者需要實質之I/O時間以完成一時間片，因為該使用者和其時間片在使用者啟動I/O操作之後為懸置可觀時間等待I/O操作完成。所以，I/O限制使用者需要長於1毫秒時間片為相當長的真實時間以完成其執行時間片以及一旦I/O限制使用者提升到執行串列300上方時，將在完成其時

五、發明說明（ 21 ）

間片之前保留在該處一段可觀時間以及往下偏移。相反地，"CPU限制"使用者需要少數I/O時間以及接近連續1毫秒或是稍長時間完成其時間片。所以，CPU限制使用者為經常往下偏移以及一般常駐在接近調度串列底部。

因此，在任何時刻，I/O限制使用者和其他具有非常大共用之使用者傾向於位於接近調度串列上方而CPU限制使用者和其他具有非常小共用之使用者傾向於位於接近調度串列底部。在二因素之中，共用和I/O相對於CPU限制，I/O量相對於CPU限制在調度串列定位上具有較大效應(對共用之正常範圍而言)。以上述方式在執行串列上使用者之定位等於高通量，因為當I/O限制使用者變成備妥執行時，亦即最後I/O操作完成時，該使用者將位於接近執行串列300上方且將快速獲得一CPU。

在圖5中，這些偏移顯示為在左方往下掃略箭頭。一些偏移為較長或是較短，因為不同使用者具有不同共用。在一典型情況中，在現行ATOD(暫態)標示之下的所有使用者為CPU-限制，而在現行ATOD(暫態)標示之上的所有使用者為I/O-限制。由於上述效應，使用者被以此效應加以排序。亦且，調度器一般將不需要在現行ATOD標示太下方尋找以發現備妥使用者，因為接近該處之CPU-限制使用者幾乎能夠總是被調度。因此在圖5左側之該"偏移"箭頭在ATOD標示或是標示之下一般傾向於由ATOD標示或是標示之下的使用者起源。然而，在調度串列上部，在ATOD標示之上，箭頭傾向於由任意處起源。請注意當一I/O限制使用

五、發明說明（ 22 ）

者完成其時間片時，因為其典型高優先順序和在先期時間片之時間的高位置，經常不移動至串列底部，以及經常仍然保留在ATOD標示之上。因此，I/O限制使用者在I/O等待完成之後將再次立即被調度，以及維持高通量。

前述排程器"非常困難放入"在吸收其共用上具有困難度的使用者。如這些使用者中之一使用者下降至其指定共用之後越來越遠處時，在調度串列中上升越來越高直到最終，假使此使用者具有任何希望取得其完全共用為止將如此執行。因此前述排程器不僅僅給予各使用者所需存取一CPU，亦且CPU各使用者多數所需共用。所需共用為不用計算各使用者消耗歷史而提供。這是因為"歷史"常駐在使用者調度串列位置中。在調度串列位置中使用者相對於ATOD標示之位置代表使用者過去已經接收之相對於使用者該得的處理器時間共用。假如使用者現在已經得到所該得時，則使用者將被定位在ATOD標示。假如使用者在ATOD標示之上，則使用者已經接收少於使用者所該得。假如使用者在ATOD標示之下，則使用者已經接收多於使用者所該得。

在步驟204中決定之所需小數R'如下文在步驟206中加以正規化。具有3個正規化案例和各案例之2變化。在3個所有案例中，計算為依據在調度串列(SRMRTHRU)上所有使用者或是在ATOD標示之下的那些使用者(SRMCTHRU)之所需小數和。在第二和第三案例中，為如上文所說明加以強制限制，VMDBK為依據被發現是否達到其(硬性或是軟性)限制而不同地考慮。在調度串列中所有使用者R'之和表

五、發明說明（ 23 ）

示為：

$$SRMRTHRU = SRMRTHRN + SRMRTHRL + R'$$

其中SRMRTHRN為在調度串列中所有使用者所需小數之和，該使用者為不限制或是尚未達到其限制，而SRMRTHRL為在調度串列中已經達到其限制的所有使用者所需小數之和，其中顯示正在加之R'代表目前正考慮之一使用者的R'，計算時尚未被加入至SRMRTHN或是SRMRTHL。類似地是，只有在ATOD標示下之使用者的和為：

$$SRMCTHRU = SRMCTHRN + SRMCTHRL + R'$$

其中SRMCTHRN為在ATOD標示下之使用者的所需小數之和，該使用者為不限制或是尚未達到其限制，而SRMCTHRL為在ATOD標示下之使用者的所需通量和。(這些變數之各變數的文字"C"為在ATOD標示下之使用者可以視為CPU限制之提醒)。在下文之計算中(所有3個案例)，只有顯示SRMRTHRN和SRMRTHRL變數，然而，假使正檢視之使用者為在ATOD標示下之的使用者，注意變數SRMRTHRN和SRMRTHRL意圖替代是重要的一件事。此考慮先前提到之3個案例中各案例的2個變化。

第一案例：

五、發明說明（ 24 ）

假使((SRMRTHRL + SRMRTHRN + R') > 1) AND SRMRTHRL > 0.5)，則某些使用者之共用必須形成為較小以及因為使用者達到其限制，所以形成大部分(>0.5)，此時它們全部包含在簡化中。之後，現行使用者正規化如下：

正規化R'=R'/(SRMRTHRL + SRMRTHRN + R')

因此，現行使用者之共用變成較小。

第二案例：

假使((SRMRTHRL + SRMRTHRN + R')>1)但SRMRTHRL < 0.5)，則某些使用者之共用必須形成為較小。但是使用者達到其限制不會太大量(它們的R'和<0.5)，所以它們不被正規化。之後只有假使現行使用者不達到其限制時，現行使用者之共用調整如下。

正規化R'=R'*(1-SRMRTHRL) / (SRMRTHRN + R')

因此，不達到其共用之現行使用者共用變成較小。然而，已經達到其共用之現行使用者使用下式：

正規化R'=R'(限制)

其中R'(限制)為特定硬性限制或是軟性限制之正規化小

五、發明說明（ 25 ）

數，亦即為合適串列時間效應調整之硬性限制或是軟性限制。因為正規化小數限於已經達到其限制之使用者，所以只要具有至少一使用者尚未達到其限制或是具有至少一未限制使用者時，此情形限制該使用者之共用在其個別限制上。

第三案例：

假使((SRMRTHRL + SRMRTHRN + R') < 1.0)，則某些使用者之共用必須形成為較大。因為已經達到其限制之使用者的共用不應該形成為較大，所以只為尚未達到其限制之使用者調整。假使現行使用者尚未達到其限制，其共用調整如下：

正規化R'=(與上文之案例(2)相同)。

但是此時正常使用者的共用變成較大(不類似上文之案例(2))。然而，假使現行使用者達到其限制時，其共用調整如下：

正規化R'=R'(限制)

因為正規化小數限於已經達到其限制之使用者，所以只要具有至少一使用者尚未達到其限制或是具有至少一未限制使用者時，此情形限制該使用者之共用在其個別限制上。

五、發明說明（ 26 ）

第一和第二案例不同之原因為假使可能時，較佳為不減低受限制使用者之共用，因為該使用者之共用從未增加(參考第三案例)。相反地，未限制之使用者共用的漏失將在另外時間得到較大共用而統計地組成。

在區別第一和第二案例時，測試SRMRTHRL > 0.5。現行使用者之R'在二案例中不考慮，因為該使用者之共用形成為較小，非較大。所以未必達到其限制，所以，當測使試以檢視該使用者和是否>0.5時，該使用者不被視為達到其限制。

在第一案例中，各R'為除以所有R'之和。因此形成值(正規化R')之和將為1。該陳式(SRMRTHRL + SRMRTHRN + R')代表所有R'之和。

回想SRMRTHRL和SRMRTHRN為SRMRTHRU之組成部分(或是陣列元素)而SRMRTHRU為R'之和。除了在上述案例執行計算的點以外，現行使用者之R'恰巧為由SRMRTHRU減去以及尚不能決定何元素將新的R'加回去。所以陳式(SRMRTHRL + SRMRTHRN+R')明確地顯示R'正加上。

在第二案例中，R'只為尚未達到其個別限制的使用者調整，然而已經達到其個別限制的使用者之R'保持常數。因此尚未達到其限制之使用者之R'藉由除以尚未達到其限制(SRMRTHRN+R')的所有使用者之R'的和以及之後將結果乘以(1-SRMRTHRL)而加以正規化。後者之因素為所需，因為尚未達到其限制的使用者之正規化共用不該加為1，因為亦具有已經達到其限制的使用者。因為正規化R'藉由只

五、發明說明（ 27 ）

除以(SRMRTHRN+R')以及省略乘數而獲得，將使值的和為1，遵守在已經乘以(1-SRMRTHRL)之後將加入至(1-SRMRTHRL)之規則。包含在SRMRTHRL的R'值事實上已經設定為等於該限制，R'(限制)。

對下列原因而言，過量之處理器時間為與其絕對或是相對共用成比率地為可使用於尚未達到其軟性或是硬性限制之所有使用者。圖式方式，此情形與在調度串列中和ATOD標示以相同速率前進(理想方式)之使用者發生。通常，先前之機構試圖因此藉由設定(或是調整)偏移而與絕對和相對共用成比率地共用處理器時間。

對在ATOD標示下之使用者而言，此比率由步驟206中R'之前述正規化的二特徵形成。如上文步驟206中說明所表示，在ATOD標示下之各使用者的所需小數R'之正規化為依據排除在ATOD標示上之使用者的在ATOD標示下之該組使用者之和SRMCTHRU。所以，此情形將傾向於給予在ATOD標示下之使用者較高優先順序，由較小偏移形成，因為"正規化R'等式"之分母為較小，亦即，當總體限制為只有那些在ATOD標示下之使用者時，各使用者的共用出現為總體之較大百分比。同時，雖然在TOD標示上之I/O限制使用者緩慢使用其時間片，但確實使用一些處理器時間。所以，因為小於100%之CPU服務將遞送至在ATOD標示下之使用者，以及其偏移現在為依據得到100%，所以在ATOD標示下之使用者將傾向於緩慢地往上移動以及事實上開始落在ATOD標示之後。然而，落在ATOD標示之後因下列原

五、發明說明（ 28 ）

因而將傾向於在調度串列中再次往下移動該使用者。使用者一經移動至ATOD標示上時，就不在ATOD標示下之該組使用者中；所以，SRMRTHRU，非SRMCTHRU，將使用在為"正規化R' "等式之分母中。此移動至ATOD標示上將傾向於在調度串列中再次往下移動該使用者，因為"正規化R' "等式之分母之後將為較大(造成較小之正規化R'，但是較大之偏移)。移動至ATOD標示上和ATOD標示下之此二傾向的淨效應為"過剩"服務將與安裝定義之絕對和相對共用成比率地分發(至能夠使用之使用者)。這些使用者以和ATOD標示相同的速率往前推進，而非較快。

位於ATOD標示上方高處之使用者為提供其能夠使用之多的處理器時間，但是它們通常不能使用過量之共用；甚至不能使用其原始共用之完全量。然而，假使其條件改變而以致於變成CPU限制時，它們因為上文表示之原因而將迅速降至ATOD標示下，以及之後藉由依據其安裝定義之共用成比率地給予它們過量處理器時間的先前原理加以管理。

步驟206之共用正規化的前述說明解釋一硬性限制或是軟性限制使用者將如何限制於其個別限制，假設具有(在ATOD標示下)至少一其他使用者尚未達到其限制或是具有至少一未限制使用者。下文說明一硬性限制使用者防止接收大於其硬性限制之一共用，以及說明在所有使用者已經到達其個別限制之後剩餘的所有過量處理器時間以最大之共用分配給軟性限制使用者。

如上文所表示，當一使用者完成一小時間片時，其在調

五、發明說明（ 29 ）

度串列中往下移動在步驟160計算之一偏移。但是在移動之前，做成一決定為此是否將移動通過最大降落量限制。各使用者之最大降落量限制等於使用者偏移之4倍，亦即，超越在ATOD標示4偏移之前。假使未限制或是軟性限制之使用者由於其個別偏移而將移動通過該最大降落量限制時，其往下移動至其最大降落量限制且不超越。該最大降落量限制使未限制和軟性限制之使用者能夠得到多餘其共用之共用，因為將造成在調度串列中定位在高於其另外支配共用之處。事實上，當所有使用者已經達到其限制以及不具有未限制之使用者時，具有最大共用之軟性限制使用者實質上將得到所有剩餘之過量處理器時間，因為該使用者將具有最小偏移，所以將具有最小最大降落量限制。此情形將把該使用者定位在調度串列中其他軟性限制使用者之上以及給予該使用者能夠利用的所有過量處理器時間。

然而，雖然當降落在最大降落量(MAXFALL)限制之下時，硬性限制使用者將不為最大降落量(MAXFALL)限制，該硬性限制使用者將由執行串列移動至限制串列。該限制串列依據使用者優先順序(VMDLPRTY)加以排序。所以當硬性限制之使用者降落在最大降落量(MAXFALL)限制之下時，通常將不再完成任何任何小時間片或是不再更進一步在調度串列中往下移動。此情形將防止硬性限制使用者獲得其特定限制之任何過量處理器時間，因為該調度器不調度在限制串列中的使用者(而一使用者在一限制串列中時不被允許進入一執行串列)。

五、發明說明（ 30 ）

在使用者已經花費足夠時間之後由該限制串列移除，如此該使用者不再佔用排程。此情形完成如下。第二"人工日時"鐘"ATOD2"被形成。此ATOD2時間將與給予使用者之具成效CPU服務和等待時間成比率地前進。當ATOD2達到限制串列頭之使用者限制串列期限時，該使用者將由限制串列移除以及放回至執行串列。為給予使用者時間以在期限到達時完成一時間片，該使用者有權在其期限之前的一時間片離開限制串列。同時存在之ATOD將繼續使用以建立使用者進入該調度串列之啟動調度串列優先順序，以及計算該最大降落量限制(MAXFALL)。換句話說，ATOD作用為調度串列和執行串列之"期限"。

偶爾使用者將經歷過長之I/O等待，例如由於在能夠繼續之前需要人介入之I/O裝置。結果，使用者將被定位在調度串列上非常高處，在其他使用者之上。當使用者開始再次執行時，這是有問題的因為假如使用者變為CPU限制，則所有的其他使用者將被防止執行直到該使用者移動通過其他使用者，且假如使用者為在非常高處以及假如正規偏移機構與將該使用者往下移動有關時將花費太長時間。所以，下列機構為用以偵測該使用者以及將其往下偏移至足以定位在調度串列上高處但尚未經歷此過量I/O等待之其他使用者鄰近處。無論何時一"非懷疑"使用者(亦即尚未懷疑為過量I/O限制之使用者)完成一時間片，該使用者如上述往下移動一偏移。然而，假如使用者在此移動期間未通過另一"非懷疑"使用者，該移動之使用者被指定為"懷疑"。

五、發明說明（ 31 ）

(此藉由設定與該移動使用者相關之一位元而完成)。下次此懷疑使用者完成一時間片時，該懷疑使用者往下移動一正規偏移以及假如該使用者在此移動期間通過一"非懷疑"使用者時，則該移動之使用者再次被指定為"非懷疑"；否則，該移動之使用者更進一步往下移動恰好在該移動之使用者遇到的第一"非懷疑"使用者之上為止。另外，在ATOD標示下之所有使用者為"非懷疑"。

一"懷疑"使用者為已經被既定TOD-TIED(TT)屬性之使用者。TOD-TIED機構之目的為不重新配置優先順序，而是如正常般保持使用者之相對順序，以及減低因上升太高所造成之任何間隙。TOD-TIED(TT)屬性將屬於在ATOD或是ATOD下之所有使用者。無論何時VMDBK重新定位將其跨過具TT之VMDBK時，其亦由一VMDBK往上(由ATOD)傳播至另一VMDBK。假使在ATOD上之VMDBK不跨過具TT之VMDBK時，則失去該屬性。在其次一小時間片之後，假使VMDBK不能夠得回TT，則其被往下強制到恰好在具TT屬性之下一VMDBK之上(或是ATOD，假使不具有介入的使用者)。

一過量I/O限制使用者難得按照預計時間結束其小時間片，因為正等待I/O完成。該使用者出現為落在排程之後，因為其請求I/O操作而非因為系統太忙以致於無法允許足夠之CPU資源給使用者。排程器44將自然地傾向於以較高優先順序將使用者排程，因為該使用者像是落在排程之後。TOD-TIED 28界限為保持防止過量I/O限制使用者使調度優

五、發明說明（ 32 ）

先順序35太高。假如使用者有時保持過量之I/O限制，則其優先順序將成為越來越高。當此情形為，使用者的優先順序為很高以及此使用者可能獨佔系統CPU資源，假如使用者下次調度時將為CPU限制。為避免此方案，當一使用者之調度優先順序35落在ATOD 23之後時取得TOD-TIED屬性，高優先順序。從那時起，具有此TOD-TIED屬性之使用者將以其在優先順序為相當接近ATOD之調度串列上叢集在一起的方式被既定，所以假使它們變成CPU限制時將不能夠接管系統。此TOD-TIED，或是"懷疑"機構為用以保持使用者調度之優先順序為在高優先順序側上之ATOD的合理位移內，而最大降落量為用以保持使用者侷限在ATOD低優先順序側的機構。

ATOD時鐘和可使用之可利用CPU時間成比率地往前推進。系統時間(管理負擔)對使用者工作或是共用為不可利用，所以不計算前進的ATOD。各執行CPU規則地增加ATOD；等待CAUS不增加。此在步驟120和122由排程器驅動。ATOD自上次增加後依據已經由多數真實CAUS.遞送之多少使用者時間而往前推進。也就是說，各CPU和該CPU的實際使用者執行時間累加器(PFXUTIME)成比率地增加ATOD。此意圖(當所有CAUS.有結果地運行)為ATOD近似即時(牆壁時鐘時間或是TOD)往前推進。當所有N CAUS.正運行時，各CPU通常為接近即時將其本身之PFXUTIME往前推進。所以N CAUS.之貢獻為除以N。

五、發明說明（ 33 ）

ATOD = ATOD + (PFXUTIME增量)/N

其中"PFXUTIME增量"為自上次檢查後此CPU之PFXUTIME累加器的變化。

ATOD2包含各CPU之使用者時間(PFXUTIME)以及各CPU之等待時間(PFXTOTWT)：

ATOD2 = ATOD2 + (PFXUTIME增量+PFXTOTWT增量)/N

其中PFXUTIME和PFXTOTWT二者為類似CPU計時器之雙字元時鐘。

然而，當任何CPU正等待多數逐次間隔時，則另一執行CPU將代表形成等待時間之貢獻至ATOD2之作用(以及在一些環境下亦貢獻至ATOD)。假設此種等待CPU正花費其所有時間在等待狀態中以及，對等待狀態而言，其貢獻至ATOD2必須與真實時間相同(如由TOD時鐘量測)。然而，另一CPU沒有做成任何假設或是貢獻於該等待CPU之執行時間。當該等待CPU再次開始執行時，將使其本身以正常方式貢獻至ATOD以及依據即時之最終可能的小數間隔貢獻至ATOD2(另一CPU已經管理先前間隔)。

如上文表示，當具有至少一CAUS.等待多數逐次時間片間隔時，一些等待時間往ATOD計算，以及同時現行CPU沒有任何工作在其執行串列中。假使現行CPU具有工作在其執行串列中，則似乎該等待CAUS.不為"可利用"以執行該

五、發明說明（ 34 ）

工作，否則它們將極有可能已經在執行該工作。相反地，假使現行CPU沒有任何工作在其執行串列中時，假使其等待時間為可利用之時間時將為非常安全地計算一等待CPU。所以，當在延伸等待模式時具有至少一CAUS.以及同時現行CPU沒有任何工作在其執行串列中時，現行CPU利用下列方程式取代先前之一方程式。

ATOD=ATOD+(增量/N)+(增量/N)*TRKCT/(N-TRKCT)

其中ATOD=ATOD+(增量/N)代表"增量"為"PFXUTIME增量"簡寫之原始方程式，以及"(增量/N)*TRKCT/(N-TRKCT)"為增加以考量等待時間之項次，再次"增量"為"PFXUTIME增量"(而非"PFXTOTWT增量")簡寫。TRKCT為SRMTRKCT，CAUS.之計數目前仍然為延伸之等待模式。

假使該等待CAUS.為許可(最後)貢獻於其本身之等待時間時，該貢獻將大塊地到達。但是，使用上述方程式時，作用之CPU代表大小不大於其本身(基於CPU)貢獻之等待CAUS.貢獻。因此在第二項次中貢獻之基本大小由增量/N代表以及乘以SRMTRKCT以貢獻多數延伸-等待模式之CAUS.但是之後該結果除以(N-SRMTRKCT)，因為具有這麼多類似現行之一CAUS.(亦即不為延伸-等待模式)的其他CAUS.，類似作用之CAUS.，將類似作用之CPU做成貢獻。

五、發明說明（ 35 ）

該SRMTODSV時間戳記為現行TOD之良好近似，雖然只為間歇地更新-在每一小時間片結束(以及其他排程點)以及無論何時VMDBK形成為備妥時，亦即接著多數I/O操作。雖然SRMTODSV為夠精確於決定是否為更新ATOD之時間，一旦做成決計時，使用更精確之TOD值。

如上文所表示，一CPU之等待時間不是總是包含在ATOD中，因為其他ATOD可能較使用者前進更快以及之後上文說明之"懷疑/非懷疑"機構(作用為限制使用者可以在調度串列中多高)將使所有使用者一致地往前推進，無論其共用有多少。然而，假使系統之多數CAUS.中只有一CAUS.(或是一些)執行以及其他CAUS經常為等待時，由ATOD完全排除等待時間造成問題。在該案例中，ATOD將延遲在後以及不和真實時間成比率地往前推進。在未限制和軟性限制之使用者或是小共用、硬性限制之使用者的案例中此情形不是問題，但是可能造成大共用、硬性限制之使用者的顯著差異。這是因為具有小共用之使用者完成介於ATOD和TOD之間的適度差異為不明顯之一時間片之後，在調度串列中往下踏一大步。但是對具有大共用(相當於小偏移)之硬性限制使用者而言，ATOD的精確動作是重要的。ATOD必須在具有大共用之使用者後面以相當迅速步伐追隨，以致於該使用者不會太快超過，達到該最大降落量限制以及過早被置放在限制串列中。假使此情形被允許發生，系統將顯示下列不想要之特徵，該特徵為忙碌系統上硬性限制之使用者將得到其共用，但是在不忙碌系統上(尤其是當其單獨在

五、發明說明（ 36 ）

系統上時)將得到小於其共用之共用。此情形藉由當實質上具有等待時間以及等待時間可以被證明為存在之工作"可利用"時將等待時間包含在ATOD而加以防止。如上文所表示，當具有CAUS.等待以及現行CPU不具有任何工作在其執行串列中時，則某部分等待時間被包含在ATOD中。現行CPU代表等待CAUS.貢獻等待時間。

在尚審理中之Davison申請案案號為第S/N 08/252,864號之申請案中，觀察到等待時間之過度貢獻可能為有害的，因為可能使"懷疑/非懷疑"機構將所有使用者無論其共用而一致地往前推進。藉由使貢獻等於現行CPU正利用之至少一CPU時間量(PFXUTIME增量)，Davison教導當計數時製成全部貢獻，亦即當具有CPU-限制工作以及幾乎使用者將被置放在限制串列中，以及當CPU不是如此忙碌時做成較少貢獻。由Davison增加之此"較少貢獻"等待時間為如本發明以及將在下文說明，實質上在更頻繁發生的環境之下由所有等待時間擴大，所以達成一較佳結果。

因為在執行串列和限制串列中的使用者由不同時鐘(ATOD和ATOD2)以不同速率加以計時，當使用者移進限制串列27或是由限制串列27移出時，執行轉換。當使用者在限制串列中時，該使用者具有二優先順序--一調度串列優先順序(VMDDPRTY)35和一限制串列優先順序(VMDLPRTY)29。該限制串列優先順序由使用者調度串列優先順序演譯出。意圖為如使用者開始其在限制串列27中的"延遲"時，該使用者必須具有由ATOD2 21與由ATOD 23

五、發明說明（ 37 ）

位移相同的位移。該位移代表在使用者因為再次執行之前必須通過的延遲量，但是該延遲需要依據ATOD2時鐘量測(以允許"等待時間"完全計算)。所以該轉換為：

VMDLPRTY = ATOD2 + (VMDDPRTY - ATOD)

當該使用者進入限制串列時，完成此VMDLPRTY之計算。當該使用者離開限制串列時，該使用者在使用者期限之前必須進入執行串列一小時間片。所以(由定義)當一使用者由限制串列移動至執行串列時，在通知使用者以及使使用者移動之前，該使用者之VMDDPRTY以下文表示之補償被設定為(ATOD+TS)以稍稍延遲超過該限制串列期限：

VMDDPRTY= (ATOD + TS) – ((ATOD2 + TS) – VMDLPRTY)
=VMDLPRTY + ATOD – ATOD2

參考圖6，和圖1相連接，圖形表示陳述為程式元件，此後參考為步驟，用於製作本發明方法之較佳具體實施例，以使用者不會太早等待或是太晚由等待移除之方式在相對共用和絕對共用使用者之間將中央處理器使用加以排程。介於這些步驟之間的關係不需要如圖6所示為連續，將在下文連接表1之本發明較佳具體實施例的虛擬碼表示而加以說明。

中央處理器之使用者共用或是其他系統資源20消耗可以

五、發明說明（ 38 ）

限制為一絕對值，為一精確的最大量26，如20%之CPU資源20。在此使用中，絕對值不參考通常和此有關的數學定義，而是由來自IBM VM排程器相對和絕對共用之觀念。例如，假如5個使用者各以相對共用100定義且所有5個使用者為作用，排程器44試圖給予各使用者1/5之系統CPU資源20。假如只有其中4個使用者為作用，排程器44試圖給予作用之各使用者1/4之系統CPU資源20。它們各依據其優先順序和同時作用之所有相對共用使用者之總體優先順序得到系統之相對共用。系統之絕對共用亦可以給予一使用者。例如，一使用者可以給予系統之絕對共用20%。此意義為無論多少相對共用使用者為作用，此使用者得到20%之系統資源。該相對共用使用者為給予剩餘之成比率共用，在此案例中為80%之CPU資源。

在步驟50中，使用者以在由(a)使用者最小共用指定24，(b)當使用者消耗中央處理器(CPU)資源20時往前推進之第一人工日時鐘(ATOD)23，(c)高調度優先順序限制(TOD-TIED)28以及(d)低調度優先順序限制(最大降落量)18定義之界限內決定的值35定處理器使用之優先順序。

至於在系統資源之共用之使用者限制為在系統目錄中以特定絕對或是相對值定義。例如，VM目錄為用以定義一使用者至系統。各使用者具有一入口可以包含SHARE(共用)敘述24、26以制訂何種共用，系統資源相對或是絕對共用為此使用者能夠具有。當系統正運行時，此共用亦可以利用SET SHARE(設定共用)命令加以動態地修正。

五、發明說明（ 39 ）

使用者為藉由'在界限內定優先順序'而給予系統資源，此方式意義為使用者為依據其優先順序35而給與系統資源如CPU資源20，但是只要一新計算之優先順序不造成使用者之系統共享落在其定義之最小共用28以下或是超越其其定義之最大共用26。例如在VM排程器44之案例中，一使用者可以定義為具有最小之相對或是絕對共用24以及一最大之相對或是絕對共用26。當計算一使用者之調度優先順序值35時，這些定義形成考慮之界限。例如，假如一使用者被指定相對共用24為200以及'限制固定'絕對最大共用26為20%以及只有此使用者另一相對200使用者為作用，該'限制固定'使用者將在20%最大界限26內制定優先順序，縱使假如最大界限尚未制訂時，該使用者可以接收50%之系統CPU資源20，

'使用者最小共用'24被定義之使用者，依據該使用者制訂之相對共用為所有作用之使用者整體相對共用之比率而取得系統資源共用。該使用者最小共用制訂在該作用之使用者的VM目錄中，以及與其餘共用相同地計算(在上文說明)，但是由該目錄使用制訂之最小共用。該使用者最大共用亦為相同計算，除了依據在VM目錄中的使用者之制訂最大共用26以外。

'人工日時'(ATOD)23為系統範圍之日時鐘格式化欄位。一特別之CPU資源不能同時為工作(亦即消耗)和等待，以及該值ATOD 23在使用者(任何使用者)工作為在系統上完成時的時間內增加，如與系統管理負擔區別。系統之調度

五、發明說明（ 40 ）

優先順序計算為依據以及和ATOD值23比較。'ATOD'23和'ATOD2'21二者為人工日時鐘。其中ATOD或是ATOD2不是實際時鐘，而是由軟體以TOD時鐘格式維持的欄位以及其中時間值為累加。軟體合適地更新這些欄位。ATOD2累加系統實際運行使用者工作之時間加上系統等待之時間。ATOD累加系統實際運行使用者工作之時間以及在特定狀況之下，亦累加系統等待之時間。所以，ATOD實際為以ATOD2累加之時間子集合。

以發生在作用之等待處理而非單獨在排程器路徑之等待時間的更新(如Davison之尚在審理中的申請案，S/N 08/252,864號之案例)，是本發明較佳具體實施例的基本特徵。如將在下文討論，以作用之等待處理中的所有等待時間而非在正常排程器路徑的所有等待時間(如上文討論只加上小部分等待時間)更新ATOD之理由，為只值得在特定狀況之下包含完全的等待時間在ATOD中，一作用之等待已經至少為例如250 ms。因此，ATOD通常累加使用者工作時間。大多數系統為該情形。然而，具有特定狀況導致一些等待時間包含在ATOD。迄今，那些狀況已經非常受限制或是稀少，而且包含在ATOD之等待時間量為可忽略。如本發明，更多等待時間在更常發生的特定狀況下包含在ATOD之累加中：也就是說，在低負載情況期間(在作用之等待常式期間)，當具有限制之使用者和中央處理器例如已經為連續作用之等待250 ms或是更多。

優先順序限制35為由'高調度'和'低調度'值定義。為防止

五、發明說明（ 41 ）

一特別使用者調度優先順序35距離現行ATOD值23、排程器44線太遠，例如限制介於現行ATOD 23和使用者調度優先順序35之間的差異為稱為高調度優先順序限制(TOD-TIED)28和低調度優先順序限制(最大降落量)18之最大值。一使用者調度優先順序35將不會被限制在那些限制外側以防止該使用者之錯誤回應時間。

在步驟52中資源20消耗為使用者藉由(a)建立指定至所選擇使用者之使用者最大共用指定26，(b)提供限制串列27以及(c)提供當使用者正消耗CPU資源20以及當CPU等待時往前推進之第二人工日時鐘(ATOD2)21，而限制為界限18、28內的特定絕對值。

限制串列為使用者之資源消耗已經達到或是超越由系統目錄中之其SHARE敘述24、26或是SET SHARE命令所允許之最大絕對共用時之使用者串列27。當在此串列27上時，一使用者不由排程器44調度，所以其資源20消耗被加以限制。一使用者可以最大絕對共用或是最大相對共用加以制訂且這些限制可以為限制固定。具有其中之一類型之最大共用(絕對或是相對)的限制固定使用者可以藉由置放在該限制串列上而加以限制。亦具有和限制固定功能不同工作之限制不定能量。限制不定使用者不藉由置放在該限制串列上而加以限制。

如本發明，為確定限制固定使用者不太快置放在該限制串列上或是不太慢由該限制串列移除，當一使用者為限制之使用者以及其調度優先順序重新計算為超越最大降落量

五、發明說明（ 42 ）

(現行ATOD+N*偏移)之新調度優先順序以及該使用者達到其限制時被加入至該限制串列。當ATOD2計算為在使用者限制串列優先順序之一小時間片內時，該使用者由限制串列移除，其中N為某預先選擇值，例如N=4。

在步驟54中使用者之限制串列27優先順序被決定，以及依據第二人工日時鐘(ATOD2)21之往前前進決定是否將使用者由限制串列移除。

在步驟56中當CPU為閒置時提供進入工作32之一作用之等待連續監視器30。此'作用之等待常式'或是'作用之等待機構為在多處理系統中使用之方法，藉由此方法不具工作執行之一處理，進入迴路以及主動尋找變成可利用之工作而非載入失能(Disable)之等待狀態。此方法解除處理器必須認知工作已經由堆疊該工作之處理器堆疊至工作正被堆疊在上面之處理器堆疊的負荷。

在步驟58中優先順序為依據最大使用共用26指定至使用者，該共用為在目錄中由其限制定義26限制的使用者可利用之最大共用CPU資源。當排程器44依據只有該使用者可利用之共用指定優先順序時具有時間以及當由於其為限制之使用者時可利用之共用多於所允許之共用時具有時間。當使用者受限制(藉由呈現在限制串列27中)以及該可利用之共用多於使用者所允許之共用26時，'最大使用共用'26為用以計算使用者之調度優先順序35。

在步驟60中當低調度優先順序(最大降落量)18被使用者調度優先順序35計算之新值超越時，使用者被置放在限制

五、發明說明（ 43 ）

串列27上。調度優先順序35在不同時間為使用者計算。為決定是否使用者必須被置放在限制串列27上的檢查在使用者調度優先順序35重新計算時製成。

在步驟62中在限制串列上的停留期間為參考ATOD 23和ATOD 21加以決定。當一使用者的調度優先順序35被計算時，為將來一事件將發生的ATOD 23值之預測，該事件為使用者小時間片之結束。ATOD 23之此預測值使用為使用者調度優先順序35。假使決定使用者必須置放在限制串列27上時，ATOD2 21而非ATOD 23使用為將使用者由限制串列移除之基準。依據ATOD 23之使用者調度優先順序之後必須改變為限制串列優先順序35(離開該限制串列27之預測時間)，必須不以ATOD2 21為基準。用以執行此之計算為：限制串列優先順序=ATOD2+(調度優先順序-ATOD)。

在步驟64中決定何時限制共用使用者正競爭CPU。

在步驟66中決定何時系統資源如中央處理單元(CPU)已經連續以及在作用之等待狀態一段定義之即時間隔(不中斷地執行；也就是說，不用尋找及執行任何使用者工作32)。當進入作用之等待狀態以及和現行TOD值比較時此決定為藉由儲存真實系統TOD時鐘而做成。

在步驟68中ATOD 23增加以及ATOD2 21被更新。

如本發明較佳具體實施例，ATOD 23之此增加為在作用之等待處理30期間完成。每次經由作用之等待碼迴路，作一檢查以決定是否具有使用者在限制串列27上。假使具有使用者在限制串列上以及假使作用之等待處理已經連續執

五、發明說明（ 44 ）

行一段特計時間間隔(例如2.5毫秒)時，則ATOD 23值以自上次ATOD 23被更新之後已經通過作用之全部時間量增加。因此，假使系統為在低負載狀態(如在沒有新的工作可利用的作用之等待至少250毫秒的處理器所定義)以及具有限制之使用者在限制串列上時，全部等待時間間隔為加入至ATOD 23。

ATOD2 21之此更新藉由其呼叫排程器亦在作用之等待處理30期間完成。每次經由作用之等待碼迴路，作一檢查以決定是否具有使用者在限制串列27上。假使具有使用者在限制串列上以及假使作用之等待處理30已經連續執行一段特計時間間隔(例如2.5毫秒)時，則作用之等待常式呼叫排程器，以自上次ATOD2 21更新後已經通過作用之時間量增加ATOD2 21。當在低負載狀態時，藉由呼叫排程器以在作用之等待處理期間更新ATOD2 21，更加頻繁地檢查將使用者由限制串列移除之必要。

在步驟70中，假如CPU為連續作用之等待狀態一段包含在ATOD2 21之時間而ATOD2 21值為在限制串列優先順序35之在一小時間片內時，使用者由限制串列27移除。

因此，如本發明之較佳具體實施例，增加CPU等待時間至ATOD 23確定使用者不過早在低負載情況(閒置CPU)時置放在限制串列27上，增加靈敏度(granularity)檢查以藉由作用之等待常式等待呼叫排程器由限制串列27移除使用者，以提供限制串列停留之較佳精確度。以此方式，介於當檢查限制串列27以移除使用者和下次檢查限制串列27

五、發明說明（ 45 ）

之間的時間間隔被大大地減低。較小時間間隔意義為靈敏度增加。

限制串列27為用以當該使用者接收多於所允許(26)之CPU資源20時保持檢查中的使用者。當此情形發生時，使用者被暫時置放在限制串列27上。排程器44檢查該串列以決定是否為將使用者由限制串列27移去之時間。這些檢查視為由限制串列'檢查移除'。此'檢查'為事件驅動。在本發明之前，限制串列在排程處理和調度此使用者工作期間由作用之處理器檢查(具有使用者工作執行之那些處理器)。如本發明，作用之等待機構亦造成這些檢查。

表1陳述本發明較佳具體實施例方法解釋之虛擬碼。該虛擬碼為參考圖2括號[]中的步驟加以註解。和表1相連參考圖1，調度器40(稱為HCPDSPCH)在行2開始由調度器迴路加以製作，監視器30在行18開始由作用之等待常式加以製作，以及排程器44在行36開始由排程優先順序重新計算碼加以製作。在行20，具有新的工作執行-例如，進入之工作32不是空白。在行25中，250毫秒之參考可加以調諧為不同系統之合適預選時間值。在行11和27中，自上次等待時間被加入後之等待時間被加入至ATOD以及藉由在行32呼叫調度器而加入至ATOD2。等待時間在等待常式期間藉由呼叫調度器而加入之理由是本發明的關鍵。先前，等待時間從未加入至ATOD2直到行11處理發生為止。在依低負載情況中，直到該處理發生可能為很長一段時間。迄今，此情形已經導致在限制串列27上的使用者保持在串列上太長

五、發明說明（ 46 ）

的時間。使用此發明，較小之等待時間在低負載情況為更頻繁地增加以及該限制串列同時被檢查，造成應該離開限制串列之使用者在必須離開時離開。表1，行35以圖1之線82表示，具有額外意義為作用之等待連續監視器30去到調度器40以當作用之等待連續監視器30增加等待時間至ATOD時更新ATOD2。表1之行50提到的新優先順序為使用者重新計算之調度優先順序(亦即一期限)，該優先順序為用以決定何時使用者將被再次調度。此調度優先順序由最大降落量之限制和TOD-TIED加以限制。排程器44在行58使用此優先順序以按照優先順序放置使用者在調度串列37上。在表1之行54等式中的數字"4"代表實驗性定義之最佳數字，以及可以在不脫離本發明範疇下加以改變。在表1之行61中，使用者依據其新計算之調度串列優先順序35在調度串列37上重新排順序。

表1：排程演算法

```
1       排程演算法:
2       HCPDSPCH(排程迴路之開始):
3       IF沒有工作將調度THEN
4       去到作用之等待常式.
5       ELSE
6       執行優先順序系統工作(亦即作用系統管理負
7       擔，非使用者工作).
8       IF一使用者之小時間片剛結束
```

五、發明說明（ 47 ）

```
9       THEN 呼叫排程器以重新計算
10      該使用者之調度優先順序
11a     以使用者時間將ATOD往前推進 [步驟 68]
11b     以使用者時間和等待時間將ATOD2往前推進.
12      假如ATOD2 在該使用者之限制串列優先順序
13      之一小時間片內時時，
14      將使用者由限制串列移除.                [步驟 70]
15      使用該調度串列以依據調度串列優先順序
16      實際執行使用者工作.
17      去到HCPDSPCH(調度器迴路之結束);
18      作用之等待常式處理      :              [步驟 56]
19      FINDWORK (迴路之開始):
20      IF具有新工作執行時THEN去到
21      調度器以執行工作項目.
22      ELSE
23      IF具有使用者在限制串列上               [步驟 64]
24      AND作用之等待處理已經
25      執行250毫秒THEN                        [步驟 66]
26      DO
27      增加等待時間至ATOD.                    [步驟68]
28      設定指示器，以致於
29      調度器將更新ATOD2
30      以及檢查該限制串列以
31      將使用者移除.
```

五、發明說明（ 48 ）

32 去到調度器HCPDSPCH
33 End
34 END
35 去到FINDWORK(迴路之結束);
36 排程器優先順序重新計算:
37 依據在使用者目錄入口中制訂的相對或是
38 絕對共用和其他登錄在非待用
39 (等待串列之)
40 使用者上的共用計算
41 系統之使用者共用.
42 IF該使用者之共用超越使用者所允許之最大共用
43 THEN使用該最大共用於
44 "偏移"計算.
45 依據使用者共用、小時間片大小
46 之大小以及依據該使用者在
47 前一時間片上之超限運轉
48 的調整計算使用者
49 之"偏移"
50 新的優先順序被決定為等於
51 舊優先順序加上剛計算
52 之偏移 [步驟50,58]
53 IF使用者新的優先順序超越最大降落量
54 (現行ATOD+4*偏移)AND此使用者為一
55 限制之使用者AND該使用者以其最大共用

五、發明說明（ 49 ）

56 排程(亦即正達到其限制)

57 THEN

58 將使用者增加至限制串列. [步驟52, 54,60, 62]

59 ELSE

60 將使用者以優先順序

61 重新排序在調度串列上.

62 end;

在先前技藝之上的優點

本發明之一優點為提供一種排程器系統和方法，以加強允許使用者在低負載情況中存取系統資源接近或是等於其硬性限制。

本發明之另一優點為提供一種排程器，加強系統資源之最佳化管理以及減低使用者在該資源管理量或是可使用性上的不安。

本發明之另一優點為提供一種系統和方法，因此監視資料使用為系統使用的排程演算法之一回饋機構以決定使用者優先順序以及決定接下來何使用者必須被許可加以執行。

本發明之另一優點為提供一種系統和方法，用於不會太快排程系統資源給藉由置放該使用者在限制串列上而具有限制存取之使用者，以及不會太慢將使用者由限制串列移除。

本發明之另一優點為提供一種系統和方法，用於不會太慢排程系統資源給藉由使資源能夠排程而具有限制存取之使

五、發明說明（ 50 ）

用者，以及不會太快禁止資源之排程。

另一具體實施例

將欣賞地是，雖然本發明具體實施例已經為解釋目的而說明，但是各種不同修正可以在不脫離本發明精神和範疇下加以製作。特別是，提供一種程式儲存或記憶體裝置如固態或是流體傳輸媒體，磁性或是光學接線，磁帶或是磁碟，或是類似物用於儲存由如本發明方法及/或其元件如本發明系統的結構控制電腦操作之機器可讀取的信號為在本發明範疇內。

另外，該方法之步驟可以在一般電腦之任何電腦如IBM系統390、AS/400、PC或是類似電腦中以及依據由任何程式語言如C++、Java、Pl/1、Fortran或是類似語言所產生之至少一或是至少一部分程式模組或物件來執行。以及另外，各該步驟或是製作各該步驟之一檔案或物件或類似物可以由特殊目的硬體或是為該目的設計之電路模組加以執行。

因此，本發明保護範疇只由下列申請專利範圍及其等效條款加以限制。

A5
B5

四、中文發明摘要（發明之名稱：　　）

用以將系統資源排程之系統及方法

一種電腦系統分配處理器時間至多數使用者。系統操作員或是其他管理員為各使用者制訂處理器時間分享至電腦。特別使用者的CPU使用在「調度驅動」多處理系統中經由建立在作用的等待常式中之監視器的使用而限制為一個絕對值。使用之機構係為一使用者串列，其CPU資源必須限制，所以他們的消耗不超過限制值。該限制串列為監視之作用等待，以決定何時使用者必須在低負載情況下由該串列移除，以及因此假使CPU可以使用時，遞送最大CPU使用至使用者。

英文發明摘要（發明之名稱：　　）

SYSTEM AND METHOD FOR SCHEDULING SYSTEM RESOURCES

A computer system allocates processor time to multiple users. A systems operator or other administrator specifies to the computer a share of processor time for each user. A particular user's CPU usage is limited to an absolute value in a 'dispatch driven' multiprocessing system through the use of a monitor built into an active wait routine. The mechanism used is a list of users whose CPU resource must be limited so that their consumption does not exceed the limit value. The limit list is active wait monitored to determine when a user should be removed from the list in a low load situation and thus deliver the maximum CPU usage to the user if it is available.

六、申請專利範圍

1. 一種由一使用者將系統資源之使用予以排程之方法，包括下列步驟：

維持一第一時鐘；

維持一第二時鐘；

在一調度處理期間，

假使有工作將執行時，只有當該使用者不在等待狀態時，將系統資源之使用依據調度優先順序排程給該使用者；否則

將該第一時鐘往前推進一使用者時間；以及

將該第二時鐘往前推進一使用者時間和等待時間；以及

在一等待處理期間，

假使有工作將執行時，呼叫該調度處理；

否則，計時等待時間；以及

假使在等待時間超越一預定值時有一處在等待狀態的使用者時，則增加第一時鐘一該等待時間和呼叫該調度處理。

2. 如申請專利範圍第1項之方法，更包括下列步驟：

在排程器處理期間，

決定該使用者之使用者共用；

假使該使用者共用超越一預先定義之最大允許共用時，依據該使用者共用和在一先前時間片上的任何超限運轉之調整時間片來計算使用者偏移；

計算等於現行調度優先順序加上該使用者偏移的該

六、申請專利範圍

使用者新調度優先順序；

假使該新調度優先順序以該使用者偏移函數超越該第一時鐘時，置放該使用者為等待狀態以及呼叫該調度處理；否則，呼叫該調度處理；以及

在調度處理期間，當該使用者完成一時間片時，呼叫該排程器處理。

3. 如申請專利範圍第2項之方法，更包括下列步驟：

甚至在調度處理期間，

假使該使用者為等待狀態以限制優先順序在該第二時鐘之一時間片內時，將該使用者由限制狀態移除。

4. 一種用於在多數受限制使用者間將系統資源之使用予以排程之方法，包括下列步驟：

維持一第一日時鐘的時間；

維持一第二日時鐘的時間；

執行一調度常式；

執行一作用之等待常式；

執行一優先順序再計算常式；

該優先順序再計算常式包含下列步驟：

計算一第一使用者之一使用者共用；

假使該使用者之共用超越一預先定義之最大允許共用時，依據該使用者共用和在一先前微小時間片上的任何超限運轉所調整之時間片大小來計算使用者偏移；

計算等於舊優先順序加上該偏移之該使用者調度優先順序；

六、申請專利範圍

假使該使用者調度優先順序超越一等於該第一時鐘加上該使用者偏移之函數的最大落差量值，以及此為一受限制之使用者，以及此使用者正以其最大共用排程時，則將該使用者增加至一限制串列中；

否則，以優先順序將該使用者記錄在調度串列上；以及

執行該調度器常式；

該調度器常式包含下列步驟：

假使沒有工作可調度時，則執行該作用之等待常式；否則，執行任何優先順序之系統工作；以及之後

假使該使用者的小時間片剛結束時，執行該排程器常式；否則，

將該第一日時鐘之時間往前推進一使用者時間；

將該第二日時鐘之時間往前推進一使用者時間和等待時間；

假使在該限制串列上之該使用者為在該第二日時鐘之一小時間片內時，將該使用者由限制串列移除；以及

依調度串列優先順序調度使用者；以及

該作用之等待常式包含下列步驟：

假使有新工作將執行時，則執行該調度器常式；否則，

假使有使用者在該限制串列上以及作用之等待處理已經超越一預先定義之限制時，則

增加等待時間至該第一日時鐘；以及

六、申請專利範圍

執行該調度器常式；

因此使用者不會太早增加至該限制串列以及不會太晚由該限制串列移除。

5. 一種將系統資源排程之方法，包括下列步驟：

(1)在多個界限內將處理器使用之使用者以優先順序處理，該等界限由下列方式界定：

(a)使用者最小使用，(b)當使用者正消耗系統資源時往前推進之一第一人工日時鐘，(c)一高調度優先順序限制，以及(d)一低調度優先順序限制；

(2)將使用者消耗之中央處理器單元加以限制為該界限內的特定值，該限制是以以下方式完成的：(a)建立指定至所選擇使用者之最大限制共用指定、(b)提供一限制串列、以及(c)提供當使用者正消耗系統資源以及當該系統資源正等待時往前推進之一第二人工日時鐘；

(3)決定限制串列優先順序以及依據該第二人工日時鐘之往前推進決定何時由該限制串列移除；

(4)當該CPU為閒置時，提供進入工作的作用之等待相互作用連續監視器；

(5)依據最大使用共用，指定優先順序給使用者；

(6)當該低調度優先順序限制被超越時將使用者置放在該限制串列上；

(7)將該第一人工日時鐘轉換為該第二人工日時鐘決定在該限制串列上的停留期間；

(8)認知何時該限制共用使用者正爭奪CPU資源；

六、申請專利範圍

(9)認知何時該系統資源已經在連續作用之等待狀態一段定義之真實時間區間以及由此以等待時間回應更新該第一人工日時鐘和該第二人工日時鐘；以及

(10)在低負載情形下，當該第二人工日時鐘值以實質上包含在該第二人工日時鐘內的所有等待時間超越該限制串列優先順序時，將使用者由該限制串列移除。

6. 一種由一機器可讀取之程式儲存裝置，實際地具體實施機器可執行之指令程式以執行用於將系統資源之使用排程給使用者的方法步驟，該方法步驟包括：

維持一第一時鐘；

維持一第二時鐘；

在一調度處理期間，

假使有工作將執行時，只有當該使用者不在等待狀態時，將系統資源之使用依據調度優先順序排程給該使用者；否則

將該第一時鐘往前推進一使用者時間；以及

將該第二時鐘往前推進一使用者時間和等待時間；以及

在等待處理期間，

假使有工作將執行時，呼叫該調度處理；否則，

計時等待時間；以及

假使在等待時間超越一預定值時有一為等待狀態的使用者時，以增加第一時鐘一該等待時間和呼叫該調度處理。

六、申請專利範圍

7.一種製造物品，包括：

一電腦可使用之媒體，具有電腦可讀取之程式碼裝置具體實施在其中以用於將系統資源之使用排程給使用者，在該製造物品中的該電腦可讀取之程式裝置包括：

用於使電腦將維持一第一時鐘起效應之電腦可讀取程式碼裝置；

用於使電腦將維持一第二時鐘起效應之電腦可讀取之程式碼裝置；

用於使電腦在調度處理期間起效應之電腦可讀取之程式碼裝置；

假使有工作將執行時，只有當該使用者不在等待狀態時，將系統資源之使用依據調度優先順序排程給該使用者；否則

將該第一時鐘往前推進一使用者時間；以及

將該第二時鐘往前推進一使用者時間和等待時間；以及

用於使電腦在等待處理期間起效應之電腦可讀取之程式碼裝置；

假使有工作將執行時，呼叫該調度處理；否則，

計時等待時間；以及

假使在等待時間超越一預定值時有一為等待狀態的使用者時，以增加第一時鐘一該等待時間和呼叫該調度處理。

8.一種用於將將系統資源之使用排程給使用者之系統，包

六、申請專利範圍

括：

一第一時鐘；

一第二時鐘；

一調度處理，用於：

假使有工作將執行時，只有當該使用者不在等待狀態時，將系統資源之使用依據調度優先順序排程給該使用者；否則

將該第一時鐘往前推進一使用者時間；以及

將該第二時鐘往前推進一使用者時間和等待時間；以及

一等待處理，用於：

假使有工作將執行時，呼叫該調度處理；否則，

計時等待時間；以及

假使在等待時間超越一預定值時有一為等待狀態的使用者時，以增加第一時鐘一該等待時間和呼叫該調度處理。

9. 如申請專利範圍第8項之系統，更包括：

一排程器處理，用於：

決定該使用者之使用者共用；

假使該使用者共用超越一預先定義之最大許可共用時，依據該使用者共用和依據該使用者共用和在一先前時間片上的任何超限運轉之調整時間片計算使用者偏移；

計算等於現行調度優先順序加上該使用者偏移之該

A8
B8
C8
D8

六、申請專利範圍 91.6.4修正補充

使用者之新調度優先順序；

假使該新調度優先順序以該使用者偏移函數超越該第一時鐘時，置放該使用者為等待狀態以及呼叫該調度處理；否則，呼叫該調度處理；以及

該調度處理，在該使用者完成一時間片時，呼叫該排程器處理。

10. 如申請專利範圍第9項之系統，更包括：

該調度處理更進一步用於：

假使該使用者為等待狀態以限制優先順序在該第二時鐘之一時間片內時，將該使用者由限制狀態移除。