無線傳感器網路中之單一相依事件排序機制

Simplex Event Ordering in Wireless Sensor and Actor Networks

吳亦超　　段裘慶　　謝政勳

Y.-C. Wu　　C.-C. Tuan　　C.-H. Hsieh

國立臺北科技大學　電腦與通訊研究所

t5419001@ntut.edu.tw　　cctuan@ntut.edu.tw　　t6418013@ntut.edu.tw

摘要

在無線傳感器網路環境中，所偵測的事件依其特性可分為獨立事件與相依事件兩種，所謂獨立事件係指事件本身和其他事件無相依關係。因此當目的端收到獨立事件之資訊時，便可直接作處理而不用考慮其它事件。反之，則需要考慮與其它事件的時間相依性作依序處理。目前研究大都集中在獨立事件上，對於相依事件較無太多的分析與探討。因此，本文提出單一相依事件排序(Simplex Event Ordering, SEO) 機制來解決相依事件在目的端的處理順序。此機制透過反應節點具有較大傳輸半徑與計算能力發出相依確認訊息，以確保反應節點在收到相依確認訊息時，能根據事件相依的特性作依序的處理，以達到處理順序與事件發生順序兩者的一致。

關鍵詞：無線傳感器網路、獨立事件、相依事件排序、反應節點、相依確認訊息。

Abstract

The events detected in a network are divided into the independent and dependent ones according to its characteristic. The independent events can be processed directly by the destination without considering the other events. On the contrary, the dependent events being relative to other ones must be handed out them in order. The research on independent events is popular, but the research on dependent events is less.

This paper proposed a Simplex Event Ordering (SEO) that could process the dependent events in sequence. This mechanism is based on the actor can sends the confirmed message by its large transmission radius and powerful computing capability.

This mechanism ensures the actor can process correctly in order after receiving dependent events. That is, ordering events is according to the time when the events detected.

Keywords : independent event, dependent event, simplex event ordering, actor, confirmed message.

1. 簡介

無線傳感器網路(Wireless Sensor and Actor Networks, WSAN)為一異質性無線感測網路，主要由資料彙集器(Sink)、感測節點(Sensor nodes, SN)和反應節點(Actor)組成，其中反應節點具有較大電量、傳輸半徑及運算能力之特性。

因反應節點功能之不同，無線傳感網路分為半自動(Semi-Automatic)與全自動(Automatic)兩種模式。

在半自動模式下，SN 將資訊傳送給資料彙集器，資料彙集器再傳送命令給反應節點，

反應節點在收到資料彙集器的命令後，會依此命令作適當處理；在全自動模式下，感測節點將資訊傳送給適當的反應節點，反應節點可直接作適當處理，而不須透過資料彙集器來發送命令。

在無線感測網路 (Wireless Sensor Network, WSN)中，大部份的研究都集中在路由建立與節能上，鮮少根據事件的排序問題加以探討，然而在某些情境下，所產生的事件多有其相依性。因此，如何根據事件的相依性，作適當的處理是本論文的主要研究重點。

在無線感測網路中，因感測節點功能有限，若要確認每個節點有無欲先處理之事件，必會延遲處理事件而造成處理順序錯誤，例如：在無線感測網路中，有一百個事件依序產生，在正常情況下，事件會依照當時發生的順序依序傳至資料彙集器，資料彙集器再依據接收的順序依序處理。然而可能因為節點本身或外在環境因素，導致事件抵達的順序和當時發生的順序不同，而造成不可預期的錯誤。因此本論文提出在無線傳感器網路的全自動模式下，以反應節點為叢集首來建置叢集樹，透過反應節點發送的相依確認訊息確保事件執行順序與發生順序相同。

在論文的第二節中，將介紹事件排序機制之相關策略，第三節介紹單一相依事件排序機制，第四節為模擬分析，最後是結論及未來工作。

2. 文獻探討

為探討事件相依處理之議題，在[1]中，作者以無線感測網路為架構提出 TMOS (Temporal Message Ordering in Sensor Networks)。TMOS 首先利用 NN (Nearest Nodes) 演算法建立一環狀拓樸，連結具有相依性質事件之感測節點及 Sink，這些感測節點感測到事件 m_i 後，複製一份同樣的複本 m_i'，並依 m_i 傳遞的反方向傳送一份 m_i'，當資料彙集器收到第一個事件後，依照封包內所含之時間標記作依序排序，直到收到與第一個事件相同之複本後，便將此事件交由後端系統處理。如圖 1 所示，SN_1 感測到 m_3，複製產生 m_3'，將 m_3 順時針傳遞至資料彙集器，m_3' 逆時針傳遞，如此在此環所產生之事件，便會在 m_3' 傳回資料彙集器前傳回，以確保所有事件皆已傳回至資料彙集器。

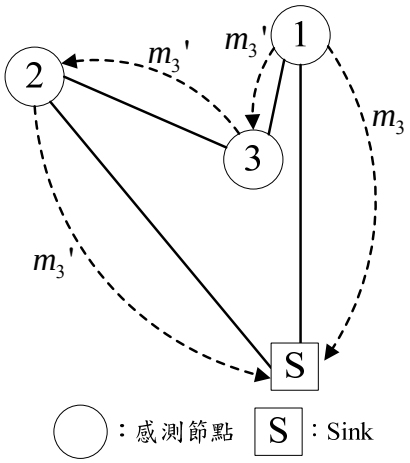


圖 1 TMOS 機制

但 TMOS 只適合在少事件及低密度節點環境下，且需浪費較多的電量與時間，因此，在[2][3][4][5]中，另一作者以無線傳感器網路為架構，提出 OBC (Ordering by Confirmation) 機制來改善 TMOS 的電量與時間，此機制以反應節點為叢集首，透過 PEQ[6]建立叢集樹狀拓樸，並利用反應節點發送同步訊息，將叢集內的感測節點同步化。當感測節點感測到事件後，會將事件傳回至反應節點，當反應節點接收到第一個事件後，便發送只含有此事件時間標記[7][8]與封包 ID 之確認訊息 m_i^* 至各末端節點，再傳回至反應節點，當反應節點收到所有鄰近節點傳回之 m_i^* 後，便依以下兩個情況將事件分別存入兩個緩衝區(buffer)內，再依事件之產生順序交由反應節點依序處理。

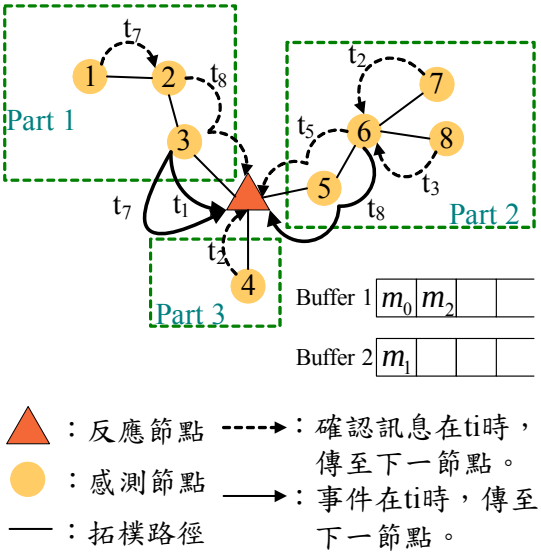
OBC 可分為兩種情況作討論，第一種情

況，當 SN 感測到 m_0，並傳回反應節點時，依此事件發送 m_0^* 至各末端節點，各路徑 m_0^* 在回傳時，由於各節點的 buffer 採 FIFO[8]機制，若此時，m_0^* 未經過的路徑上之 SN 產生 m_1，則 m_1 必會在 m_0^* 傳回前，先傳遞至反應節點，並將 m_1 存入 buffer 1 與 m_0 排序比較，待反應節點的所有鄰居節點傳回 m_0^* 後再將 buffer 內的事件依序處理，因此，可確保事件處理的順序性。另一種情況，當各路徑的 m_0^* 在回傳時，有一 m_0^* 在經過路徑之 SN 才產生 m_1，這時 m_1 將會存入 buffer 2，並等待 buffer 1 中的事件處理完畢後，再針對 buffer 2 中 m_1 為依據發送 m_1^* 至各末端節點，待所有反應節點的鄰居節點將 m_1^* 傳回反應節點後，再將 buffer 2 之事件依序處理。

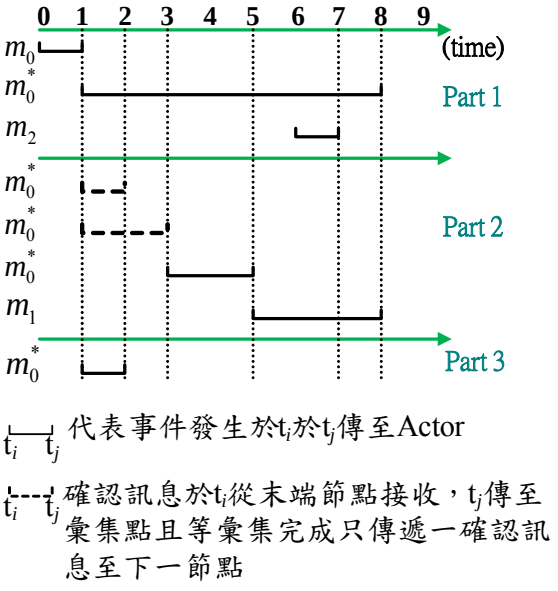
雖然 OBC 在電量消耗與傳遞時間上都比 TMOS 好，但由於事件是否存入在前確認訊息所依據之 buffer，在於前確認訊息是否經過此發生事件之節點，若經過後才發生，則存入不同之 buffer，若未經過，則存入同一 buffer。在此種情況下，當事件數超過三個以上時，便有可能發生處理順序的錯誤。

為了方便探討 OBC 範例之錯誤，以反應節點的鄰居節點將樹狀拓樸分為數個 Part，如圖 2(a)與圖 2(b)所示，因反應節點有三個鄰居節點，所以分為三個 Part 作討論，在 Part 1 中的 SN_3 感測到 m_0，在時間點 t_1 將 m_0 傳至反應節點並存入 buffer 1，再由反應節點發送 m_0^* 至各末端節點；在 t_2 時，Part 2 中的末端 SN_7 之 m_0^* 已於 SN_6 上等待且在 Part 3 中的 m_0^* 也於 t_2 傳回至反應節點；在 t_3 時，SN_8 之 m_0^* 也傳至 SN_6，此時 SN_6 才發送一 m_0^* 至 SN_5；在 t_5 時，Part 2 中的 m_0^* 也傳回反應節點，且 SN_6 也於此時感測到 m_1；在 t_6 時，Part 1 中 SN_3 感測到 m_2；在 t_7 時，m_2 傳回反應節點並存入 buffer 1；在 t_8 時，Part 1 中的 m_0^* 傳回反應節點，此時三個部分的確認訊息皆已傳回，OBC 會將反應節點的 buffer 1 中的事件依序處理，然而 m_1 此時尚未傳回反應節點，假若傳回，也是存入 buffer 2，所以當 m_0、m_2 都依序處理完後，m_1 才會被反應節點處理，造成處理事件順序錯誤。

由上述可知，OBC 雖大幅減少處理相依事件之時間與所消耗電量，但在某些特殊情況下仍無法百分百確保事件處理的順序性，因此本文提出單一相依事件排序機制來解決此問題。



(a) 事件與確認訊息傳遞路徑



(b) 時序圖

圖 2 OBC 之錯誤範例

3. 單一相依事件排序機制

單一相依事件排序機制主要以 WSAN 為架構，並透過 PEQ 建立以反應節點為叢集首的樹狀叢集，並由反應節點廣播時間同步封包，以對叢集內的 SN 作時間同步化， SN 的 buffer 處理機制則採用 FIFO 機制，讓 buffer 內的事件能依照時間標記作順序處理，以避免外部節點的事件比本身節點的事件先行處理。

為了解決 OBC 無法百分百對相依事件作依序處理問題，SEO 提出相依確認訊息機制，取代 OBC 的確認訊息機制，並將 buffer 由兩個降為一個。舉例來說，m_0 比 m_1 先發生，但可能因為某些因素造成 m_1 比 m_0 先傳送到反應節點。這時，反應節點便會先執行 m_1 的動作，造成事件處理不當。如圖 3(a)所示，火災發生時，A 區先偵測到煙霧，接著煙霧瀰漫至 B 區域，導致 B 區也偵測到煙霧，如圖 3(b)所示，這時後端系統必須知道 A 區為第一個偵測到煙霧的地區，進而對起火點作滅火動作。倘若後端系統判斷錯誤，誤認 B 區為第一個偵測到煙霧的地區，但事實上 B 區只有煙霧，並不是起火點，這時，若對 B 區作滅火動作，反而會造成火災的擴大。

3.1 相依確認訊息機制

SN 感測到事件後將之傳遞至反應節點，進入 buffer 的事件便根據時間標記排序，最早發生的事件會排在 buffer 之最前端，反之，則排於後端。反應節點再根據 buffer 內排在最後端之事件 m_L ，發送一含有與此事件相同時間標記與事件 ID 之相依確認訊息 m_L^{Δ} ，並將此訊息發送至各路徑之末端節點，再傳回反應節點，當回傳的相依確認訊息個數 k 等於反應節點之鄰居節點個數 N_{nei} 時，表示各部分之相依確認訊息皆已傳回，便將時間標記小於此相依確認訊息之事件作依序處理，若 buffer 內仍有事件，則繼續執行 SEO 直到 buffer 內沒有事件，如圖 4 所示。

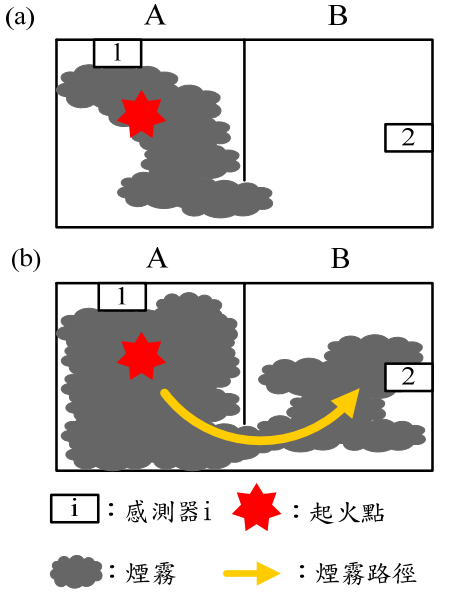


圖 3 火災煙霧瀰漫情境

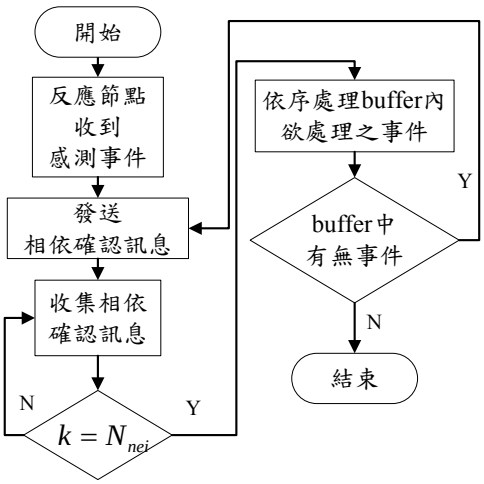


圖 4 SEO 流程圖

SEO 和 OBC 不同的地方，在於確認訊息所具備功能不同，如圖 5 所示，m_f 為第一個事件反應節點所接收之事件，在 m_f 產生的時間 $t[m_f]$ 時，Part 1 中的 SN 開始傳遞 m_f 至反應節點；在時間 $t[m_f^{\Delta}]$ 時，反應節點接收到 m_f 並同時發送 m_f^{Δ} ，在 t_1 前產生之事件稱為 m_{bf} 。因在網路初期時，第一個被反應節點接

收之事件 m_f 也在 t_1 前產生，所以 m_f 也為 m_{bf}，因 buffer 內一存有事件便開始執行 SEO 機制，此時的 m_f 為緩衝區中的第一個事件，同時也為最後一個事件即 m_L，所以反應節點會發送 m_f^{Δ} 至各末端節點，m_f^{Δ} 再回傳至反應節點。Part i 中各路徑的末端節點在 t_1 時接收到 m_f^{Δ}，當反應節點收到 Part i 所傳回之 m_f^{Δ}，可確保 m_{bf} 在 t_2 前皆已回傳至反應節點，然而在 t_1 後產生的事件 m_{af} 則無法確保，因為其事件發生時間 $t[m_{af}]$ 晚於 t_1，由此可知相依確認訊息主要功能在所有相依確認訊息傳回反應節點前，能確保小於其時間標記之事件皆已傳回，並在 buffer 內根據事件時間標記來依序處理時間標記小於 m_L 之事件。

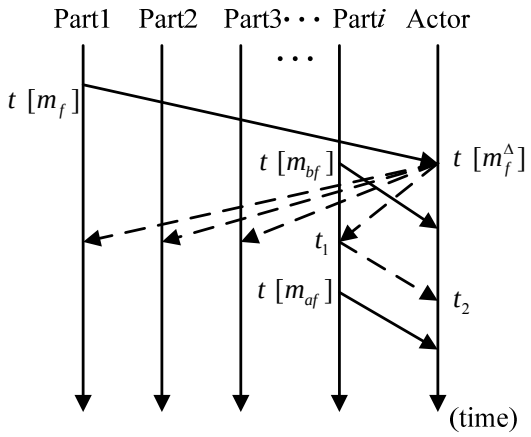
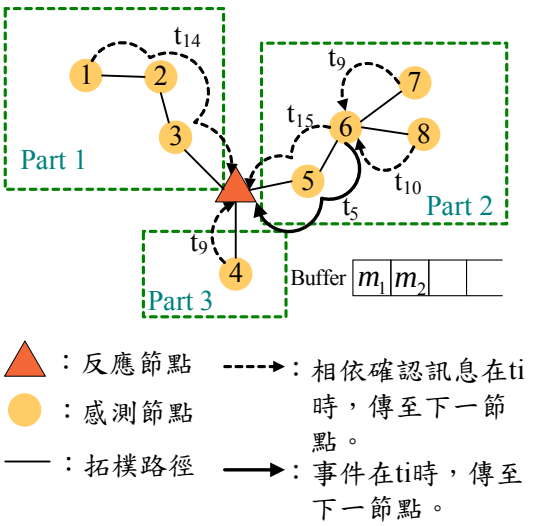


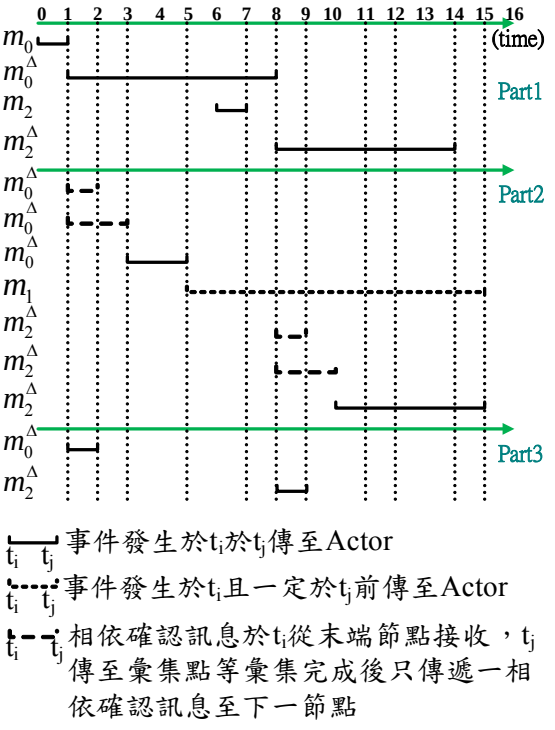
圖 5 SEO 確認先處理事件回傳時序圖

3.2 相依確認訊息與確認訊息

OBC 中的確認訊息 m_i^* 之功能，在於反應節點的所有鄰居節點將 m_i^* 傳回至反應節點後，會將此 m_i^* 之事件所在之 buffer 內所有事件依序作處理，不同於 OBC 機制的 SEO 中的相依確認訊息 m_i^{Δ}，則是在反應節點的所有鄰居節點將 m_i^{Δ} 傳回至反應節點後，只處理 buffer 中小於和等於此相依確認訊息時間標記之事件。



(a) 事件與相依確認訊息之傳遞路徑



(b) 時序圖

圖 6 SEO 改善 OBC 之錯誤範例

舉例來說，在 OBC Part 1 中的 SN_3 感測到 m_0，並在 t_1 將 m_0 傳至反應節點，反應節點再發送 m_0^* 至各末端節點；在 t_2 時，Part2 中的末端 SN_7 之 m_0^* 已於 SN_6 上等待另一 m_0^*；在 t_3 時，SN_8 之 m_0^* 也傳至 SN_6，這時 SN_6 會發送一 m_0^* 至 SN_5，同時在 Part3 中的 m_0^* 也於 t_2 傳回至

反應節點；在 t_5 時，Part2 中的 m_0^* 也傳回至反應節點，且 SN_6 也於此時感測到 m_1；在 t_6 時，Part 1 中 SN_3 感測到 m_2 ；在 t_7 時， m_2 傳回至反應節點並存入 buffer；在 t_8 時，Part 1 中 m_0^* 也傳回至反應節點，所以在 t_8 時，三個部分的 m_0^* 皆已傳回，此時，反應節點會將 m_0 與 m_2 依序作處理，然而 m_1 卻還在 buffer 中等待第二階段的確認才能作處理，如圖 2 所示。

上述情況為 OBC 所造成處理順序錯誤，以下為 SEO 解決此錯誤之範例，如圖 6 所示。而在 SEO 下反應節點會確認 buffer 內有無事件排於 m_0 之前，若沒有事件排於 m_0 之前，只處理 m_0，若有，將排於 m_0 之前的事件與 m_0 依序作處理，在 t_8 時，反應節點會發送 buffer 內排於最後之事件 m_2 的確認訊息 m_2^Δ 至各末端節點；在 t_9 時，Part3 中的 m_2^Δ 已傳回反應節點，Part2 中 SN_7 傳遞 m_2^Δ 至 SN_6；在 t_{10} 時，Part2 中的 m_2^Δ 於 SN_6 彙集，並向 SN_5 發送 m_2^Δ；在 t_{14} 時，Part1 中 m_2^Δ 傳回反應節點；在 t_{15} 時，Part2 中 m_2^Δ 也傳回至反應節點，當三個部分的 m_2^Δ 都已傳回，此時反應節點會依序處理 buffer 內時間標記小於 m_2 之事件包括 m_2 ，因此 SEO 能確保事件依序處理之正確性，如圖 6 所示。

3.3 SEO 順序之正確性証明

Definition

G：The set of the event's timestamp.

G_i is the event's timestamp of the batch i.

Lemma

因 SEO 確認訊息之功用，在於反應節點的所有鄰居節點傳回相依確認訊息至反應節點後，可確保本批處理之事件時間標記皆小於後一批處理事件之時間標記，因此批次間的順序正確。

$$\{x \mid x \in G_{i-1}\},\ \{y \mid y \in G_i\},\ 2 \le i < n,\ i,n \in N \ \wedge\ \forall x < \forall y$$

Corollary

設網路中有 n 個事件產生，分成 i 批處理，每批處理事件數為 j 件，$1 \le j \le n$ 。當 $i = n$，則 j=1，因 Lemma 可知其處理順序正確。當 $i = n-1$ ，則存在一批處理事件數其 $j = 2$ ，因兩事件會在 buffer 內經過比較，所以可知此批事件排序正確，又因 Lemma 能確保批次間的順序正確，由此可知，事件順序無誤。以此類推，直至 i=1 時，所有事件皆在 buffer 內排序，所以不論分幾批處理，皆能保證處理事件之時間標記為遞增數列。由以上推論得知，當網路中發生 n 個事件，SEO 能保證其事件處理順序正確。

4. 模擬與分析

SEO 以.Net 為平台，並使用 C++撰寫模擬程式，每一次的模擬將隨機產生 100 個事件，當所有事件皆由反應節點處理應用後，檢查其事件時間標記是否為遞增數列，是則列入成功模擬，否為失敗。模擬參數如表 1 所示，網路中 SN 每秒發生一至兩個事件直到發生一百個事件後則停止，藉此來比較 SEO 與 OBC 在處理事件之順序正確性、能量消耗以及處理事件所花費的延遲時間。

表 1 模擬參數

Parameters	Value
模擬環境範圍	60×60 (m^2)
事件數目	100 (個)
節點數目	50 (個)
模擬時間	50 (s)
產生事件之節點數	20 (個)
產生事件率 (R_m)	1~2 (events/s)
反應節點 傳輸半徑	20 (m)
傳輸耗能 (E_t)	14.88 (mW)
接收耗能 (E_r)	12.50 (mW)
One hop delay	20 (ms)
感測節點部署方式	隨機

4.1 正確排序事件率

在模擬 N 次後，檢查其成功次數 N_{correct}，再將此兩數取一比率 P_{t}，如公式(1)。比較在相同環境下，SEO 與 OBC 之 P_{t}。

$$P_{\text{t}} = \frac{N_{\text{correct}}}{N} \times 100\% \quad (1)$$

N_{correct} :事件處理順序正確之模擬次數
N :總模擬次數
P_{t} :正確排序事件率

4.2 延遲時間

SEO 與 OBC 下反應節點皆會發送確認訊息來確認事件順序，在確認完順序後，事件才會交由反應節點處理應用，這邊模擬 N 次取 SEO 下總事件傳遞時間 $\sum_{i=1}^{N} T_{\text{SEO}}^{i}$ 與 OBC 下總事件傳遞時間 $\sum_{i=1}^{N} T_{\text{OBC}}^{i}$ 之比值 $R_{\text{t}}^{\text{tol}}$，如公式(2)。若值越小表示 SEO 傳遞時間較快。

$$R_{\text{t}}^{\text{tol}} = \frac{\sum_{i=1}^{N} T_{\text{SEO}}^{i}}{\sum_{i=1}^{N} T_{\text{OBC}}^{i}} \quad (2)$$

T_{SEO}^{i} :SEO 機制事件傳遞及處理時間
T_{OBC}^{i} :OBC 機制事件傳遞及處理時間
$R_{\text{t}}^{\text{tol}}$:SEO 與 OBC 處理時間之比值

4.3 電量消耗

SEO 機制與 OBC 機制皆須要發送確認訊息，當處理事件分的批數越多時，節點傳遞的封包數量也就越多，其電量的消耗也越多，因此比較 SEO 與 OBC 發送確認訊息之次數乘上所需消耗的電量，即能評估何種方法較為節省能源，這邊取 SEO 下發送確認訊息次數 E_{SEO}^{i} 與 OBC 下發送確認訊息次數 E_{OBC}^{i} 之比值 $R_{\text{e}}^{\text{tol}}$，如公式(3)。若值越小表示 SEO 較 OBC 節能。

$$R_{\text{e}}^{\text{tol}} = \frac{\sum_{i=1}^{N} E_{\text{SEO}}^{i}}{\sum_{i=1}^{N} E_{\text{OBC}}^{i}} \quad (3)$$

E_{SEO}^{i} :SEO 下發送相依確認訊息之次數。
E_{OBC}^{i} :OBC 下發送確認訊息之次數。
$R_{\text{e}}^{\text{tol}}$:SEO 與 OBC 發送確認訊息次數比值。

5. 結論與未來工作

從論文的搜尋與研讀可以發現，如何根據事件的相依性作依序的處理是相當重要的，因此，本文以無線傳感器網路為架構，提出單一相依事件排序機制，此機制主要透過反應節點具有較大傳輸半徑與計算能力之特性和相依確認訊息，讓反應節點能根據事件發生的順序作依序的處理。在未來工作上，將持續針對 SEO 與 OBC 作模擬分析，以證明 SEO 在事件排序上之正確率，延遲時間與電量消耗，並根據部分相依、群組相依和即時處理加以深入探討，最後加入節點或連結斷裂時的容錯排序機制。

致謝

本研究獲「臺北科技大學 97 年教學卓越計畫─伯樂計劃」贊助，特此致謝。

參考文獻

[1] K. R¨omer, “Temporal message ordering in wireless sensor networks,” *In IFIP Mediterranean Workshop on Ad-Hoc Networks*, June 2003, pp. 131-142.

[2] A. Boukerche, F.H.S. Silva, R.B. Araujo ,and R. W. N. Pazzi, "A low latency and energy aware event ordering algorithm for wireless actor and sensor networks", *in Proc. of the 8th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, NY, USA, 2005, pp. 111–117.

[3] A. Boukerche, A. Martirosyan, "An efficient algorithm for preserving events' temporal relationships in wireless sensor actor networks," *in Proc. of the 32nd IEEE Conference on Local Computer Networks*, Oct. 15-18, 2007, pp. 771-780.

[4] A. Boukerche, R.B. Araujo, F.H.S. Silva, and L. Villas, "Wireless sensor and actor networks context interpretation for the emergency preparedness class of applications," *Computer Communications*, pp. 2593-2602, Sep. 2007.

[5] A. Boukerche, R.B. Araujo, and F.H.S. Silva, "An efficient event ordering algorithm that extends the lifetime of wireless actor and sensor networks," *Performance Evaluation*, pp. 480-494, June 2007.

[6] A. Boukerche, R.W.N. Pazzi, and R.B. Araujo, "A fast and reliable protocol for wireless sensor networks in critical conditions monitoring applications", *in Proc. of the 7th ACM/IEEE International Symposium on Modeling*, Oct. 4–6, 2004, pp. 157-164.

[7] E. Yoneki, J. Bacon, "Determination of time and order for event-based middleware in mobile peer-to-peer environments", *in Proc. of the 3rd Int'l Conf. on Pervasive Computing and Communications Workshops*, 2005, pp. 91-96.

[8] R. Hayton, *OASIS: An Open Architecture for Secure Interworking Services*, University of Cambridge, 1996.

[9] V.P. Silva, R.B. Araujo, "A study on the quadrilateral of fire and fire incidents," *in Proc. of the Technical Report-DC-TR 10/2005 Computer Science Department-UFSCar*, Jan. 2005.