行政院國家科學委員會專題研究計畫　成果報告

子計畫一：動態旅次起訖推估與預測(I)

計畫類別：整合型計畫
計畫編號：NSC93-2218-E-009-043-
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日
執行單位：國立交通大學統計學研究所

計畫主持人：周幼珍

報告類型：精簡報告
報告附件：出席國際會議研究心得報告及發表論文
處理方式：本計畫可公開查詢

中　華　民　國 94 年 11 月 1 日

Abstract

Real-time Origin-destination (O-D) information is important in many transportation related domains, especially in Intelligent Transportation System (ITS). The conventional ways to obtain O-D data is costly and real-time information is not achievable. To obtain real-time O-D information in a reasonable way, state space model with Gibbs sampler and Kalman filter is then introduced by researchers. The Gibbs sampler method to estimate O-D requires a huge computation time, thus computing power must be increased to match the goal of real-time information. This paper implements parallel computation on a Linux cluster for the origin algorithm and given a sample of real road network. The parallel implementation introduced in this paper lead to a satisfying result.
Keywords — dynamic origin-destination, state space model, Gibbs sampler, traffic, PC cluster

Introduction

Origin-destination (O-D) data is very important in many transportation related domains(Bernstein, 1996, 1997, 2001; Chang, 1994, 1995, 1999, 2004, 2005; Jou, 2001,2002), such as transportation planning, urban and regional planning, traffic assignment and so on. In Intelligent Transportation Systems (ITS), real-time O-D information(Jou, 2003,2004) also plays an important role in Advanced Traffic Management System (ATMS), and Advanced Traveler Information System (ATIS) to provide real-time traffic management and information. With real-time information, many high value ITS applications such as emergency vehicle routing in time shortest-path, just-in-time delivery would be feasible. The traditional way to gather O-D information includes license plate recognizing, automatic vehicle identification and so on. In reality, O-D information collection is very difficult and costly. The accuracy in comparison of license plate is low, and also the real-time information is not attainable with roadside survey. Due to the high cost of O-D data collection in highway systems, researchers have been seeking estimation methods to derive valuable O-D flow information from less expensive traffic data, mainly, link traffic counts of surveillance systems. Jou introduce state space model into dynamic O-D estimation, which estimate O-D matrices and transition matrix simultaneously without any prior information of state variables, while other studies assume that the transition matrix is known or at least approximately known, which is unrealistic for a real world network (Jou, 2003). Gibbs sampler is introduced in the solution algorithm to overcome the shortcoming of known transition matrix. Gibbs sampler is a particular type of Markov Chain Monte Carlo (MCMC) algorithm, the consumption of computation power is huge. In ITS, many applications require real-time information, in order to achieve the goal of real-time information, parallel computing techniques is introduce to improve the performance of computation. The remainder of this paper is organized as follows. The dynamic origin-destination estimation by state space model is introduced in section 2. Section 3 addresses the parallel method to the origin solution algorithm and its results. Finally, conclusions are outlined in section 4.

Dynamic Origin-Destination Estimation

State space model is introduced to estimate O-D flow from link traffic counts. The standard state space model is coupled with two parts: transition equations and observation equations. First, the state equation which assumed that the O-D flows at time *t* can be related to the O-D flows at time *t*-1 by the following autoregressive form,

$$x_t = F x_{t-1} + u_t, \quad t = 1,2,3,...,n \qquad (1)$$

where x_t is the state vector which is unobservable, F is a random transition matrix,

$u_t \sim N_p(0,\Sigma)$ is independently and identically distributed noise term, where N_p denotes the *p*-dimensional normal distribution, Σ is the corresponding covariance matrix. *x* the state variable, is defined to be the path flow belonging to an O-D pair.

Next, the observation equation,

$$y_t = Hx_t + v_t, \quad t = 1,2,3,...n \qquad (2)$$

where y_t is the $q \times 1$ observation vector which means there are *q* detectors on the road network. The number of O-D pairs is denoted by *p*. *H* is a $q \times p$ zero-one matrix, which denotes routing matrix for a network. v_t is also a noise term that $v_t \sim N_q(0,\Gamma)$. Both *x* and *F* are unobservable, thus Kalman filter is not suitable to directly estimate and forecast the state vector. Hence, Gibbs sampler is used to tackle the problem of simultaneous estimation of *F* and x_t by latest available information.

There are two major elements to be incorporated in the solution method, i.e. filtering states by observations and sampling scheme of *F* and state variables. Since the observations y_t are not used in the conditional distribution, the Kalman filter and the Gibbs sampler must be combined. The Gibbs sampler is a technique for generating random variables from a distribution indirectly, without having to calculate the density. It is a Markovian updating scheme that proceeds as follows. Given an arbitrary starting set of values $Z_1^{(0)}, Z_2^{(0)}, Z_3^{(0)},..., Z_k^{(0)}$, and then draw $Z_1^{(1)} \sim \left[Z_1 \middle| Z_2^{(0)}, Z_3^{(0)},...Z_k^{(0)}\right]$, $Z_2 \sim \left[Z_2 \middle| Z_1^{(0)}, Z_3^{(0)},...Z_k^{(0)}\right]$, and so on. Each variable is visited in the natural order and a cycle requires *k* random variate generations. After *i* iterations we have $(Z_1^{(i)}, Z_2^{(i)}, Z_3^{(i)},..., Z_k^{(i)})$. Under mild conditions, Geman and Geman showed that the following results hold(Robert, 1998).

Result 1 Convergence

$$(Z_1^{(i)}, Z_2^{(i)}, Z_3^{(i)},..., Z_k^{(i)}) \to \left[Z_1, Z_2, Z_3,..., Z_k\right] \text{ and hence for each } s,\ Z_s^{(i)} \to [Z_s]$$

as $i \to \infty$. In fact a slightly stronger result is proven. Rather than requiring that each variable be visited in repetitions of the natural order, convergence still follows any visiting scheme, provided that each variable is visited infinitely often.

Result 2 Rate

Using the sup norm, rather than the L_1 norm, the joint density of $(Z_1^{(i)}, Z_2^{(i)}, Z_3^{(i)},..., Z_k^{(i)})$ converges to the true density at a geometric rate in *i*, under visiting in the natural order.

Result 3 Ergodic theorem

For any measurable function *T* of $Z_1, Z_2, Z_3, ..., Z_k$ whose expectation exits,

$$\lim_{i\to\infty} \frac{1}{i} \sum_{l=1}^{i} T(Z_1^{(l)}, Z_2^{(l)}, Z_3^{(l)}, ..., Z_k^{(l)}) \to E\left(T\left(Z_1, Z_2, Z_3, ... Z_k\right)\right)$$

As Gibbs sampling through *m* replications of the aforementioned *i* iterations produces *k* tuples $\left(Z_{1j}^{(i)}, Z_{2j}^{(i)}, Z_{3j}^{(i)}, ..., Z_{kj}^{(i)}\right)\left(j = 1,2,3,..., m\right)$, which the proposed density estimate for $\left[Z_s\right]$ having form $\left[\hat{Z}_s\right] = \frac{1}{m} \sum_{j=1}^{m} \left[Z_s \middle| Z_r^{(j)}, r \neq s\right]$.

The above Gibbs sampling scheme on a random transition matrix and state variable forms the center part of the algorithm. In the process of generate state variables, Kalman filtering mechanism is added. The solution algorithm is shown as follows,

- *Step 1 (Initialization)*
  1. Use prior information to generate $F^{(0)}$
  2. Given Σ and Γ
  3. Given $x_0 \sim N(\mu_0, V_0)$
- *Step 2 (Generate* $x_t^{(g)}, \quad t = 0,1,2,..., n$ *)*
  1. Generate $x_0^{(g)}$ from $N\left(\mu_0, V_0\right)$
  2. Generate $x_1^{(g)}$ from $x_1 \middle| x_0^{(g)}, F^{(g)} \sim N\left(F^{(g)} x_0^{(g)}, \Sigma\right)$
  3. Use the Kalman filter to filter $x_1^{(g)}$
  4. Repeat 2, 3 for *t* = 2,3,…,n
- *Step 3 (Generate* $F'^{(g)}$ *)*
  1. Calculate $A^{(g)} = \left\{a_{ij}^{(g)}\right\}$, $a_{ij}^{(g)} = \left(X'^{(g)}_{n(i)} - X'^{(g)}_{n-1} \hat{F}'^{(g)}_i\right)' \left(X'^{(g)}_{n(j)} - X'^{(g)}_{n-1} \hat{F}'^{(g)}_j\right)$

     and $\hat{F}'^{(g)}_i = \left(X^{(g)}_{n-1} X'^{(g)}_{n-1}\right)^{-1} X^{(g)}_{n-1} X'^{(g)}_{n(i)}$
  2. Calculate $X^{(g)}_{n-1} X'^{(g)}_{n-1}$
  3. Generate $w \sim Wishart\left(X^{(g)}_{n-1} X'^{(g)}_{n-1}, n - p\right)$
  4. Generate $Z = \left(z_1', z_2', z_3', ..., z_p'\right), z_k \overset{iid}{\sim} N_p\left(0, A^{(g)}\right)$

5. Generate $F'^{(g)} = \left(\left(w^{\frac{1}{2}} \right)' \right)^{-1} Z$

- *Step 4 (Iteration)*
  Repeat *Step 2, Step 3 m* times, then we have $\{X^{(1)},..., X^{(m)}\}$.
- *Step 5 (Estimate X and F')*
  Repeat *Step 1* to *Step 4 k* times, then we have $\{X^{(m)}{}_{(1)},..., X^{(m)}{}_{(k)}\}$. Finally, estimate *X* and F' by $\hat{X} = \frac{1}{k}\sum_{n=1}^{n=k} X^{(m)}{}_{(n)}$ and $\hat{F}' = \frac{1}{k}\sum_{n=1}^{n=k} \hat{F}'^{(m)}{}_{(n)}$.

The Implementation of Parallel Computation and its Results

To achieve real-time information requirement, computing power is critical. In order to achieve this goal, parallel computing is then introduced. The solution algorithm can be divided into several independent computation parts by dividing it at step 5. Given *n* computation nodes and *k* times of iterations, each node will take care of $\frac{k}{n}$ iterations. Each process store its own $X^{(m)}{}_{(n)}$ and $\hat{F}'^{(m)}{}_{(n)}$, when the number of iterations is reached, all of them is then gathered together to estimate *X* and F'. In this situation, communication between computing nodes is minimum, and computing power can be easily increased without communication bandwidth limitation. Figure 1 describes the parallel architecture. In the pre-processor section, parameters used in our algorithm are initialized, so does the necessary input data. When assign jobs, these input data are sent to computing nodes in the cluster through TCP/IP base intranet with Message Passing Interface (MPI) Library. The computational procedure for the parallel process consists of:

Step 1. Load input data and parameters. Initialize MPI environment.

Step 2. Count the computing nodes exits in the cluster environment. Decide the count of samples should be generated by each computing nodes. Send data to each computing nodes.

Step 3. Each computing nodes generate its own $X^{(m)}$ and $F'^{(m)}$ by given input data for given times. And then send the result to server.

Step 4. After all the data been sent to server, the server estimate $\hat{X}$ and $\hat{F}'$ by $X^{(m)}$ and $F'^{(m)}$ samples from each computing nodes.

Step 5. Stop MPI environment. Output data.

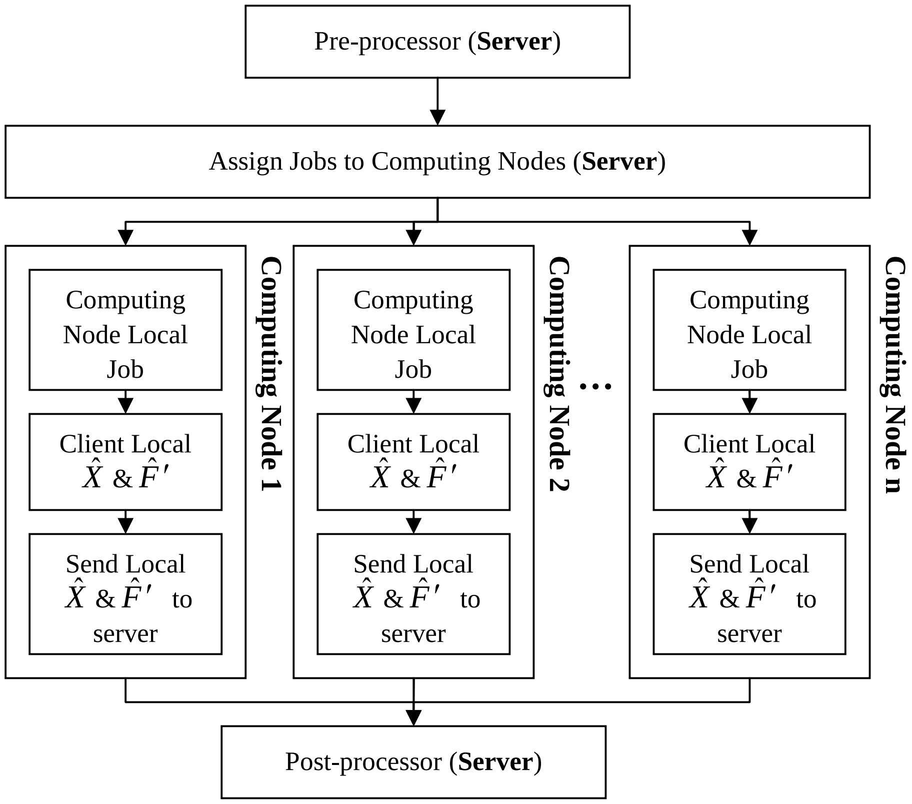

Figure 1. The flow chart of parallel algorithm

The result shown below is a real network in Hsinchu Science Park in Taiwan with 8 observation sites and 48 links. The observation of traffic count updates every minute, 30 time intervals of observation data is used to estimate the O-D flow. The number of Gibbs sampler iteration *m* is fixed to 500, and the number of samples to estimate X and F' is floating. The result is presented in Table 1, the unit of time is second.

	Parallel Computation Time (seconds)		
	Samples		
Number of processors	k = 100	k = 50	k = 25
2	1137.86	573.49	312.34
4	576.43	287.54	157.526
8	289.76	152.09	81.762
16	152.7383	84.516	44.454
32	92.26	49.26	20.35

Table 1. Parallel Computation Time comparison

The parallel environment of this research consists of 16 computing nodes; each contains 2 Intel XEON 3.2GHz processors and 1 GB memory. Nodes are connected with a 1Gbits 3Com gigabits Ethernet switch for MPI protocol and a 100 Mbits PCI fast Ethernet switch for Network File System (NFS) and Network Information System (NIS). Figure 2 shows the speedups and efficiencies, where the speedups is the ratio of the code execution time on a single processor to that on multiple processors and efficiency is defined as the speedup divided by the number of processors(Gropp, 1999; El-Rewini, 1998), of the parallel computing for 100 samples on the 32 CPU Linux-cluster with MPI library. As shown in Figure 2, a quite good value of the speedup and efficiency of the parallel scheme is achieved. That means we can decrease the computation time easily to achieve the goal of real-time information.

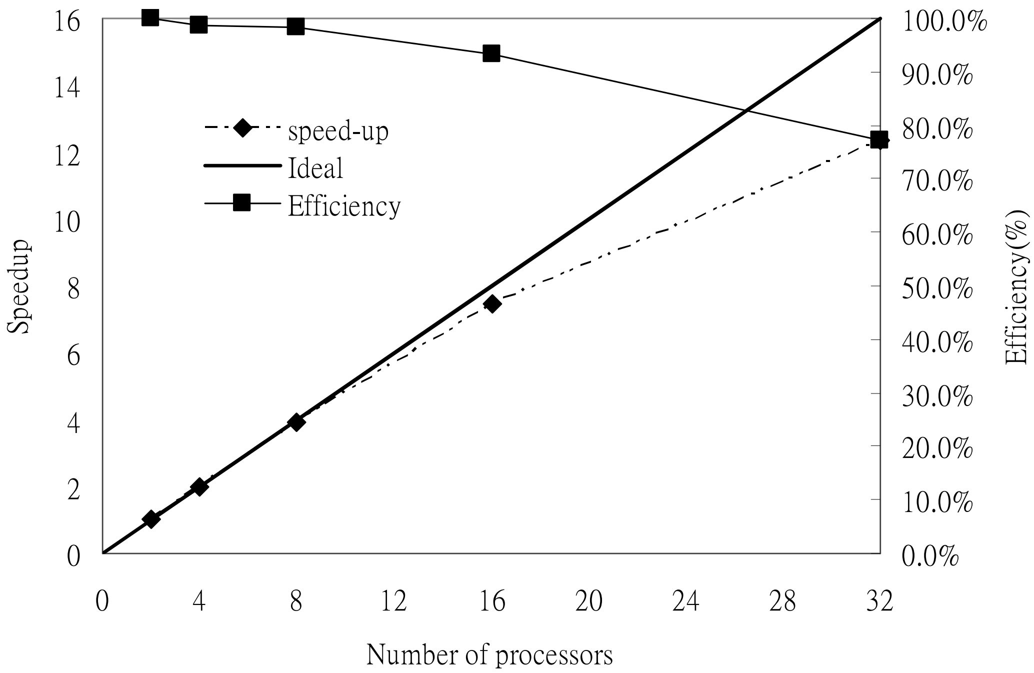

Figure 2. Speedups and efficiencies for the parallel computing of k = 100

Conclusions

This paper provides a parallel implementation on a PC-based Linux cluster with MPI library to estimating origin-destination matrices for general road network by using the state space model with Gibbs sampler and Kalman filter. With the experiment of real network data, the parallel implementation presented in this paper is efficient and can increase the computing power easily to match the goal of real-time information.

References

[1] Christian P. Robert, "*Discretization and MCMC Convergence Assessment*", Springer, ISBN: 0-387-98591-3, 1998

[2] D.A. bader, J. JaJa, R. Chellappa, "*Scalable data parallel algorithms for texture synthesis using Gibbs random fields*", IEEE Transactions on Image Processing, Volume: 4, Issue: 10, 1995

[3] Daniel Sorensen, "*Likelihood, Bayesian and MCMC methods in quantitative genetics*", Springer-Verlag, 2002

[4] David Bernstein, T.L. Friesz, Z. Suo and R.L. Tobin, "*Dynamic Network User Equilibrium with State-Dependent Time Lags*", Networks and Spatial Economics, Volume: 1, pp. 319-347, 2001

[5] David Bernstein, T.L. Friesz, "*Infinite Dimensional Formulations of Some Dynamic Traffic Assignment Models*", Network Infrastructure and the Urban Environment, Springer, New York, pp.112-124, 1997

[6] David Bernstein, T.L. Friesz and R. Stough, "*Variational Inequalities, Dynamical Systems and Control Theoretic Models for Predicting Time-Varying Urban Network Flows*", Transportation Science, Volume: 30, pp.14-31, 1996

[7] Gang-Len Chang, Xianding Tao, "*Advanced computing architecture for large-scale network O-D estimation*", Proceedings of the 6th 1995 Vehicle Navigation and Information System Conference, pp.317-327, 1995

[8] Gnag-Len Chang, J. Wu, "*Recursive estimation of time-varying origin-destination flows from traffic counts in freeway corridors*", Transportation Research 28B, pp.141-160, 1994

[9] Gnag-Len Chang, X.D. Tao, "*An integrated model for estimating time-varying network origin-destination distributions*", Transportation Research 33A, pp.381-399, 1999

[10] H. El-Rewini, T. G. Lewis, *"Distributed & Paralle Computing."*, Manning, Greenwich, NY, 1998

[11] Iljoon Chang, Gang-Len Chang, "*Calibration of Intercity Trip Time Value with a Network Assignment Model: A Case Study for the Korean NW-SE Corridor*", Journal of Transportation and Statistics, Volume: 5, 2002

[12] John H. Mathews, "*Numerical Methods*", Prentice-Hall, ISBN: 0-13-626656-8, 1987

[13] Pei-Yih Ting, R.A. Iltis, "*Diffusion network architectures for implementation of Gibbs samplers with applications to assignment problems*", IEEE Transactions on Neural Networks, Volume: 5, Issue: 4, 1994

[14] Pei-Wei Lin, Gang-Len Chang, Saed Rahwanji, "*Robust Model for Estimating Freeway Dynamic Origin-Destination Matrix*", Transportation Research Board,

2005

[15] Pierre Bremaud, “*Markov chains: Gibbs fields, Monte Carlo simulation, and queues*”, Springer, 1999

[16] W. Gropp, E. Lusk, R. Thakur, *“Using MPI-2: Advanced Features of the Message-Passing Interface.”*, MIT Press, Cambridge, MA, 1999

[17] Yow-Jen Jou, Ming-Chorng Hwang and Jia-Ming Yang, “*A Traffic Simulation Interacted Approach for the Estimation of Dynamic Origin-Destination Matrix*”, IEEE International Conference on Networking, Sensing and Control, Volume: 2, 2004

[18] Y.J. Jou, M.C. Hwang, Y.H. Wang, and C.H. Chang, “*Estimation of Dynamic Origin-Destination by Gaussian State Space Model with Unknown Transition Matrix*,” 2003 IEEE International Conference on Systems, Man & Cybernetics, 5-8 Oct. 2003, Washington, D.C., USA.

[19] Yow-Jen Jou, Ming-Chorng Hwang, “*Rapid Transit System Origin-Destination Matrix Estimation*”, 2002 World Metro Symposium & Exhibition, Taipei, April 2002

[20] Yow-Jen Jou, “*Rapid Transit System O/D Pattern Calculation with Statistical Gibbs Sampling and Kalman Techniques*”, Conference on Computational Physics 2001, 5-8 September, 2001, Germany