2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

A Parallel Computing Framework for Air Traffic Flow Management

Yi Cao and Dengfeng Sun

Abstract— **The nationwide air traffic flow control for the National Airspace System is a complicated large-scale optimization problem which is significant to the future development of Air Traffic Management. Based on a Link Transmission Model and dual decomposition method, the large-scale air traffic flow optimization is decomposed into smaller independent optimization subproblems and solved using parallel computing. As the model is solved with a Mixed Integer Linear Programming, searching for an optimal integral solution usually entails longer runtime than Linear Programming. To improve the applicability of this model, a parallel computing framework is developed, which explores the parallelism of the model in order to increase the computational efficiency. Heterogeneous computers are clustered in a Client/Server topology to carry out parallel computing. By further exploring the multithreading capability of multi-core computer, the optimization task is distributed to multiple processors to process in a parallel fashion. Simulation shows that the parallel computing framework decreases the runtime of nationwide air traffic optimization from hours timescale to minutes timescale. Moreover, compared to conventional single thread optimization method, the framework can achieve expected runtime reduction by deploying more computer resources without worrying about the increased complexity of the traffic network, therefore making the near real-time air traffic flow optimization possible.**

I. INTRODUCTION

In the next decade, there will be a significant increase in air transportation demand in U.S. by a factor of two to three according to the forecast by Joint Planning and Development Office (JPDO) [1]. Such a growth inevitably stresses efficient utilization of the limited airspace which is one of the critical issues associated with the goal of the Next Generation Air Transportation System (NextGen). According to the blueprint of NextGen, all aircraft and airports in U.S. will be connected to the NextGen network and continually share information in real time. A National Airspace System (NAS)-wide air traffic optimization platform provides a decision-making support for the Air Traffic Controller with regard to efficient allocation of airspace resources, therefore will improve the airspace utilization efficiency and benefit the decision-making process. On the other hand, the NAS-wide air traffic optimization also provides an evaluation platform for Air Traffic Management (ATM) research, such as airport operations, aircraft holding policy, en route air traffic control.

However, modeling the behavior of the multicommodity flow network could be challenging. There are over fifty thousand aircraft flying in the NAS of the U.S. each day, and over five thousand every minute during the rush hours. There are two classes of paradigms in the field of ATM, namely Lagrangian Models and Eulerian Models. The former are trajectory-based model while the latter are aggregate model [2], [3], [4]. The Lagrangian Models precisely propagate the trajectory of individual aircraft, so the dimension of the problem is proportional to the number of aircraft involved in the model. As a result, trajectory-based modeling generally leads to intensive computation if large number of aircraft are considered. In contrast, the Eulerian Models focus on the aggregate properties of the air traffic network [5], therefore have lower fixed dimensions. As a tradeoff, the Eulerian Models are not able to provide detailed information about each individual aircraft in general. To accommodate the requirements at both microscopic and macroscopic level, a Large-Capacity Cell Transmission Model (CTM(L)) was proposed in [6]. It is formulated as a linear discretized dynamical system whose dimension is independent of the number of aircraft. By introducing control variables into the model, CTM(L) is able to model the control strategies imposed on the en route traffic flow. Another remarkable advantage of CTM(L) is that it is able to provide detailed traffic information at both sector and air route levels. However, billions of state variables are needed for NAS-wide air traffic optimization which entail computational difficulty in realistic implementation. To make the large-scale optimization problem tractable, [8] introduced a dual decomposition method to decouple the air traffic network, resulting in an iterative optimization approach. However, even a 2-hour traffic optimization may be time consuming. In an effort to improve the computational efficiency, [9] modified the CTM(L) by aggregating the aircraft count at a link level, resulting in a Link Transmission Model (LTM). Due to the decreased number of state variables by an order of ten, LTM runs faster than CTM(L).

In our previous effort, computational efficiency improvement largely depends on the modeling properties. If the traffic doubles or triples in near future as anticipated, more air routes will be added into the current NAS. The complexity of the air traffic network will increase as a result. Previous methods will fail to maintain the expected computational efficiency because of the increased runtime of optimization. Therefore, it is imperative to seek other method whose runtime is independent of the complexity of the air traffic network. Nowadays parallel computing technique is becoming more and more prevalent at different hardware levels, such as pipeline, multi-cores, multiprocessor computer. Distributed computation resources are clustered to tackle problems in fields like finite element structural analysis, computational fluid dynamics, and Monte Carlo Simulation. Most of these applications are similar in that the large-scale problems

Yi Cao and Dengfeng Sun are with School of Aeronautics and Astronautics, Purdue University. cao20@purdue.edu, dsun@purdue.edu.

are decomposed into smaller subproblems and solved by heterogeneous computer clusters using parallel computing. In ATM literature, parallel computing was also successfully introduced into solving the traffic flow problem. [10] employs a classical Lagrangian model, i.e. Bertsimas and Stock-Patterson (BSP) model, to simulate the traffic, and applies Danzig-Wolf decomposition to decrease the size of the problem. [11] proposes a parallel architecture to solve the problem presented in [10]. Higher computational efficiency is achieved via multi-threading.

There are two problems facing current TFM research. First, most models developed in historical works are validated using traffic at Center level or on limited air routes only. There are few works addressing traffic flow optimization at the nationwide level. Work presented in this paper aims to fill this gap. Second, due to the limited capability of earlier optimization tool, early works, such as [10] and [8], solves the traffic problem via Linear Programming (LP) relaxation where the solution is expected to be integral. An empirical rounding method must follow to convert the fractional variables into integers. This would cause loss of optimality, and some constraints may be broken as well [10]. Currently, many optimization tools, like CPLEX, support Mixed Integer Linear Programming (MILP) resolution. Integral solution can be obtained directly as long as the variables are defined as integer. However, MILP usually takes longer runtime to search for the optimal integral solution than LP. The runtime can be unacceptable if the size of problem is very big. To offset this negative impact, this paper proposes a parallel computing framework. By using dual decomposition method the NAS-wide air traffic optimization is decomposed into a collection of subproblems path by path based on the LTM [9]. As each subproblem is a smaller optimization problem that can be solved independently in a relatively short runtime, solving them in parallel significantly reduces the total runtime. The compute-intensive optimization problem thus can be efficiently solved. Since single machine has limited computing power, this paper develops parallel computing using multiple computers, with each computer running multiple processes. It is expected that by maximizing the computation capability of hardware higher computational efficiency can be achieved.

Although the methodology used in this paper is similar to the work presented in [10] and [11], this paper puts more emphasis on realistic hardware and software implementation. Aside from the different traffic model and decomposition method, the parallel architecture in this study is also achieved at a higher level, namely multiple computers. This work is an integration of algorithm, software and hardware.

The rest of the paper is organized as follows. Section II introduces the dual decomposition method based on the LTM which is designed for the TFM optimization. Section III elaborates the integration of hardware and software system. Section IV presents the simulation results. Computational efficiency is discussed in this section as well. Concluding remarks are provided in Section V.

II. LINK TRANSMISSION MODEL AND DUAL DECOMPOSITION METHOD

A. Link Transmission Model

Link Transmission Model is an Eulerian-Lagrangian Model. It uses the high altitude sectors as the geographic basis. Each flight path is modeled as an air route connecting departure airport and arrival airport, as shown in Fig. 1. The path is segmented by the sector boundaries, each segment is called a *link*. A sector contains several links belonging to different paths. Aircraft flying through the same series of sectors are considered in the same link. Aggregating the aircraft in the same sector forms the so-called sector count which is often used as the indicator of traffic conditions in a region. The NAS is covered by a network composed of variety of paths. When the aircraft fly across the sector boundaries, they are considered as flows transitioning between the links.

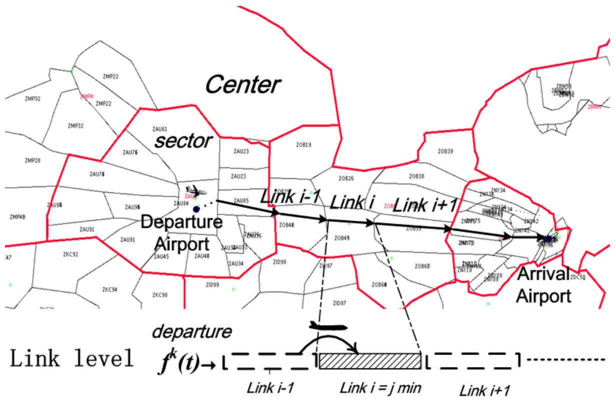


Fig. 1. Link Transmission Model

The state variable $x_i^k(t)$ is defined as the aggregate aircraft count in link i on path k at instant t, $x_i^k(t) \in \mathbb{Z}^+$. The traffic flow complies with flow conservation principle, is formulated as a discretized linear time varying system:

$$\begin{aligned} x_i^k(t+1) &= \beta_{i-1}^k(t)x_{i-1}^k(t) + (1-\beta_i^k(t))x_i^k(t), \\ x_0^k(t+1) &= f^k(t) + (1-\beta_0^k(t))x_0^k(t). \end{aligned}$$

where $f^k(t)$ is the scheduled departure into path k. $\beta_i^k(t)$ is a transmission coefficient representing the fraction of aircraft transitioning from upstream link i to downstream link $i+1$, hence $0 \le \beta_i^k(t) \le 1$. In optimization problem, optimizing $\beta_i^k(t)$ amounts to regulating the flow rate. In order to obtain a linear program, a new state variable is introduced, $q_i^k(t) = \beta_i^k(t)x_i^k(t)$. By definition, $q_i^k(t) < x_i^k(t)$. then the dynamics for path k can be rewritten in vector form:

$$X^k(t+1) = IX^k(t) + BQ^k(t) + Cf^k(t) \qquad (1)$$

where I is a $n^k \times n^k$ identity matrix. n^k is the number of links on path k. $X^k(t) = [x_0^k(t), x_1^k(t), \cdots, x_{n^k}^k(t)]^T$, $Q^k(t) = [q_0^k(t), q_1^k(t), \cdots, q_{n^k}^k(t)]^T$, and

$$B = \begin{bmatrix} -1 & & & \\ 1 & -1 & & \\ & & \ddots & \\ & & 1 & -1 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

$$Q^k(t) \le X^k(t),$$
$$q_i^k(t), \quad x_i^k(t) \in \mathbb{Z}^+.$$

Unlike the Lagrangian Models, the dimension of LTM is determined by the number of links on a path rather than the number of aircraft involved. Thus the overall dimension of NAS-wide traffic model is proportional to the number of paths identified in the system.

B. Formulation of TFM optimization

There are various ways to formulate the objective function. One of the methods is to minimize the total flight time over all aircrafts in the planning horizon, which reflects the realistic goal to minimize fuel consumption:

$$\min \sum_{k=0}^{K}\sum_{t=0}^{T}\sum_{i=0}^{n^k} c_i^k x_i^k(t) \tag{2}$$

where T is the planning time horizon. K is the number of paths identified in the NAS. c_i^k is the weight imposed on a link. Airline fairness can be adjusted by imposing different weights on the links of a flight path.

Current NAS uses *Monitor Alter Parameter* (MAP) to restrict the number of aircraft in a sector in order to guarantee safety [12]. Thus the sector count should not exceed the MAP value C_{s_i}.

$$0 \le \sum_{(i,k)\in Q_{s_i}} x_i^k(t) \le C_{s_i} \tag{3}$$

where Q_{s_i} represents the set of links lying in sector s_i.

A minimum dwell time in a link is imposed on each aircraft to ensure a reasonable flow rate. The minimum dwell time T_i is the average flight time an aircraft needs to pass through link i, which is derived statistically from historical flight data.

$$\sum_{t=T_0+T_1\ldots+T_i}^{T^*} q_i^k(t) \le \sum_{t=T_0+T_1\ldots+T_{i-1}}^{T^*-T_i} q_{i-1}^k(t) \tag{4}$$
$$T^* \in \{T_0+T_1\cdots+T_i,\ldots,T\}$$

Equation (4) reflects the fact that the accumulated inflow of a link is greater than its accumulated outflow at T_i look-ahead time instant. This amounts to detaining an aircraft in link i for at least T_i.

It is also assumed that no aircraft is held in the air at the end of the optimization, i.e. every flight will land in its destination airport. Therefore the accumulated departures must be equal to the accumulated arrivals.

$$\sum_{t=0}^{T} q_{n^k}^k(t) = \sum_{t=0}^{T} f^k(t) \tag{5}$$

Equations (1), (3), (4), (5) together with the objective function (2) formulate the TFM optimization problem. Solving this problem yields optimal flow for each flight path as well as associated flow controls, which can be used for TFM evaluation. Given that real traffic controls are generally applied to an individual aircraft instead of a flow, the flow controls generated from this model seems inexecutable. [7] develops a disaggregation method to convert the flow controls into flight-specific control actions for CTM(L), which can be easily adapted to LTM as well. But this is out of the scope of this paper.

C. Dual Decomposition Method

Equation (2) indicates that the number of state variable is determined by three indices, namely K, T, n^k. Consider a 2-hour NAS-wide TFM optimization with a time interval of one minute, which typically involves approximately 2400 paths each with 15 links on average, then there are $120 \times 2400 \times 15 = 4,320,000$ state variables. Moreover, $Q^k(t)$ is also treated as state variable at the same order as $X^k(t)$, then the number of state variables is up to 8,620,000. It is difficult to handle a problem of such a high order with current optimization tool available. However, all constraints are separable in terms of path except for Equation (3). State variables of different paths are coupled only by the sector capacity constraint. [8] introduced the dual decomposition method to decouple the paths. By introducing Lagrangian Multipliers, the sector capacity constraints are incorporated into the objective function. The formulation can be reorganized in terms of path. A detailed induction can be found in [8] and [9]. Table I summarizes the dual decomposition algorithm based on LTM. A flowchart of the algorithm is shown in Fig. 2.

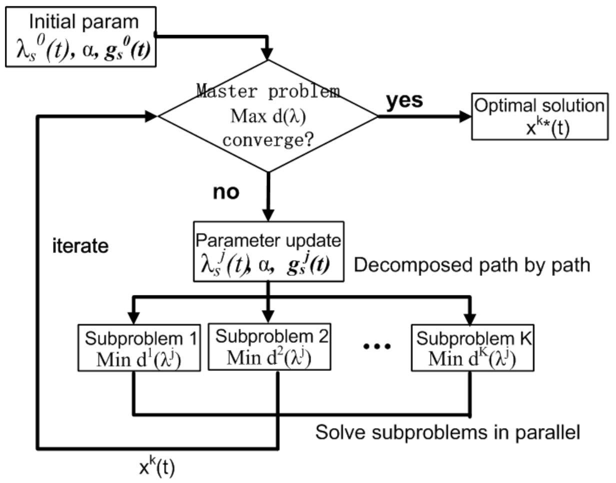


Fig. 2. Flow chart of dual decomposition algorithm

An important feature of the dual decomposition method is that it fits a parallel computing framework. Taking advantages of nowadays multithreading technique, the optimization can be allocated to distributed computer resources.

In **Step 1** of Table I, each subproblem is a smaller MILP. The number of state variables is reduced compared to original NAS-wide TFM optimization. Thus It is easy to solve by CPLEX. The prevalent optimization tool CPLEX provides high-performance APIs for efficient implementation. Most importantly, CPLEX supports multiprocessing. Computers with multiprocessor can run several optimizer instances simultaneously. This feature enables parallel computation.

TABLE I

DUAL DECOMPOSITION ALGORITHM BASED ON LTM

```
Initialization:
    $\lambda_{s_i}^0(t) = \lambda_0, X(0) = X_0$
Step 1:
    for  $k = 1 : K$ solve subproblem (for path $k$):
        $d^k(\lambda_{s_i}^j(t)) = \min \sum_{t=0}^{T} \sum_{i=0}^{n^k} [c_i^k + \lambda_{s_i}^j(t)] x_i^k(t)$
        $s.t.$
            $x_i^k(t+1) = x_i^k(t) + q_{i-1}^k(t) - q_i^k(t)$
            $x_0^k(t+1) = x_0^k(t) + f^k(t) - q_0^k(t))$
            $q_i^k(t) \le x_i^k(t), \qquad q_i^k(t) \in \mathbb{Z}^+$
            $0 \le \sum_{(i,k) \in Q_{s_i}} x_i^k(k) \le C_{s_i}, \qquad x_i^k(t) \in \mathbb{Z}^+$
            $\sum_{t=0}^{T} q_{n^k}^k(t) = \sum_{t=0}^{T} f^k(t)$
            $\sum_{t=T_0+T_1 \ldots +T_i}^{T^*} q_i^k(t) \le \sum_{t=T_0+T_1 \ldots +T_{i-1}}^{T^*-T_i} q_{i-1}^k(t)$
                $T^* \in \{T_0 + T_1 \cdots + T_i, \ldots, T\}$
Step 2:
    Update objective of master problem:
        $d(\lambda_{s_i}^j(t)) = \max\{-\sum_{t=0}^{T} \sum_{s_i=0}^{S} \lambda_{s_i}^j(t) C_{s_i} + \sum_{k=0}^{K} d^k(\lambda_{s_i}^j(t))\}$
    if  $d(\lambda_{s_i}^j(t))$ converge
        return  $X^* = X(t)$
    else
        $g_{s_i}^{j+1}(t) = -(\sum_{(i,k) \in Q_{s_i}} x_i^k(t) - C_{s_i})$
        $\lambda_{s_i}^{j+1}(t) := (\lambda_{s_i}^j(t) - \alpha_i g_{s_i}^{j+1}(t))_+$
        goto Step 1

where
    $s_i \in \{0, \cdots, S\}, t \in \{0, \ldots, T\}, k \in \{0, \ldots, K\}$.
    $g_{s_i}^j(t)$ is the subgradient of dual function.
    $\lambda_{s_i}^j(t)$ is the Lagrange multiplier, $(\cdot)_+$ denotes the non-negative
        part of a number.
    $\alpha_j = \frac{1}{j+1}$ is the subgradient step and $j$ is the iteration index.
```

III. INTEGRATION OF DISTRIBUTED AIR TRAFFIC FLOW OPTIMIZATION SYSTEM

In this section, a platform, which includes hardware and software integration, for the parallel computation is introduced.

A. Hardware topology

Based on the master problem-subproblems structure of the dual decomposition method, a Client/Server model is used, as shown in Fig. 3. In the one-to-many deployment, the workload is balanced to maximize the efficiency. The server is responsible for disseminating data, distributing subproblems to each client, receiving optimal results and updating the parameters. Each client simply concentrates on solving subproblems. The Client/Server model is connected via WAN or LAN. Internet-based distributed system provides considerable flexibility in system deployment. a) Compatibility of heterogeneous computer resources via TCP/IP or UDP/IP protocol. b) Extensibility using existing network. c) High data throughput. Therefore, Internet-based structure is widely accepted in many ATM researches.

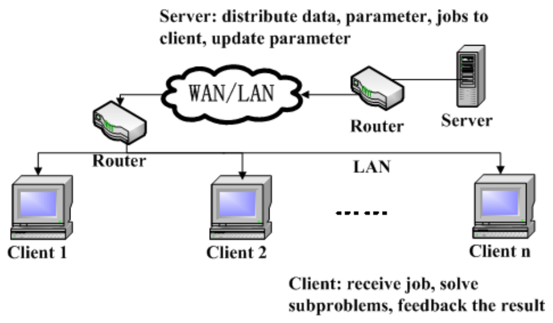

Server: distribute data, parameter, jobs to client, update parameter
WAN/LAN
Router
Server
Router
LAN
Client 1
Client 2
......
Client n
Client: receive job, solve subproblems, feedback the result

Fig. 3. System topology

B. Software

A Client/Server software architecture is depicted in Fig. 4. To accelerate the computation, both *Server* and *Client* run multiple processes with each playing different roles, e.g. communication, synchronization, solving subproblems. Details are provided as follows.

In the *Server* end, there are three processes.

- A *Communication thread* is responsible for sending and receiving data. It monitors the data transfer request from internal *Master thread* and external *Clients*. When a request is received, this module conducts data buffering. Since large data blocks are sent using UDP which is not a reliable connection, it is the *Communication thread's* responsibility to guarantee reliable data sending/receiving through error checking mechanism. Resending is issued upon request.
- A *Master thread* controls the progress of the whole optimization. It evenly divides the original problem into blocks of jobs and sends to the *Clients*. When each iteration is finished, the *Master thread* updates the Lagrangian Multipliers $\lambda_{s_i}(t)$ using the feedback of optimal traffic flow $X^*(t)$.
- A *Database thread* provides an access to the database. To set up the LTM, information regarding to links, paths and sectors are needed. All the information is extracted offline from historical flight record using data mining and stored in the database. This thread responds to inquiry from the*Master thread* and *Clients*, feedback is buffered to the shared memory or to the *Communication thread* respectively.

The *Client* has a similar configuration as the *Server*.

- A *Communication thread* is the same module as in the *Server*.
- A *Parent thread* receives a block of subproblems from the *Server* and redirects the job to the *Child thread* for processing. In this second level job distribution, the *Parent thread* does not evenly distribute the job to each *Child thread*, but allocates jobs according to the computation workload of each subproblem to achieve the workload balance. It continuously sends new jobs to those idle *Child threads*, thus maximizes the computational efficiency.
- Several *Child threads* receive subproblems from the

Parent thread and perform optimization. Given that each thread would compete for the CPU time, the number of *Child threads* should match up to the *Client's* multithreading capability. In a multi-core machine, the efficiency could be maximized if each processor is busy on executing the computation all the time.

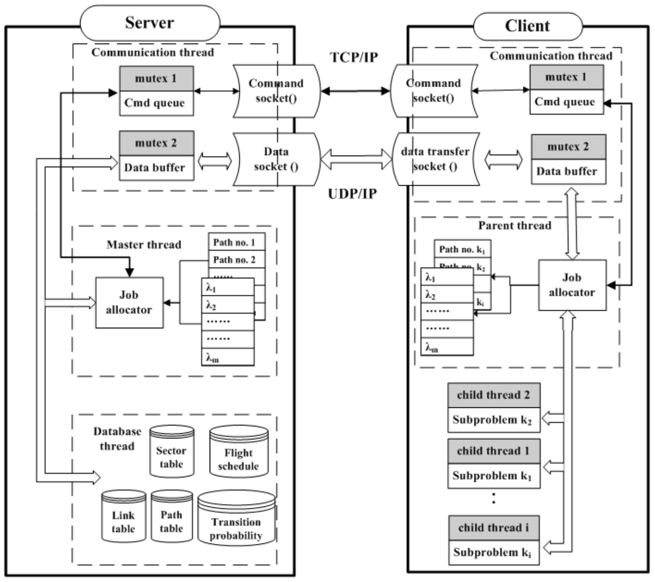


Fig. 4. Software architecture: Client/Server model

The software design has to give consideration to the following aspects.

- Data communication. In the WAN/LAN environment, data communication is via TCP/IP or UDP/IP. The communication is propagated via *Socket*. Two communication channels are established for data transmission between the *Server* and the *Client*. One is command channel, the other is data transmission channel. The command channel is used for synchronization purpose. A set of commands are defined. The Client/Server act in a request-reply mode. Since commands are short messages which must be correctly received, the command channel uses TCP/IP. The data transmission channel uses UDP to transfer large data blocks, such as the data from the database, updated parameters and optimized traffic data. In order to ensure reliable data transmission, the *Hand-shaking* is established via command channel before initiating a data transmission. Likewise, error checking must be done after each receival. Resending request is issued in case of long network latency or incorrect data receival.
- Synchronization. The *Server* is responsible for synchronizing the work of the *Clients* by sending synchronization command. Internal thread synchronization is accomplished through *mutex*, *signal* and *shared memory*. Given that the data transfer is a duplex communication, internal and external accesses may compete for the public buffer resources. *mutex* provides lock mechanism to prevent reading/writing "dirty" data. The internal communication is via *signal*. Other option such as Message Passing Interface is equivalent in terms of efficiency, but *signal* is more suitable for Object-Oriented programming when one defines the action for a thread.

IV. SYSTEM VALIDATION

In this section, a 2-hour NAS-wide air traffic is used to validate the proposed framework. Ten Dell Workstations construct a small prototype of the proposed distributed optimization system. Each workstation is configured with a 2.8 GHz 8-processors INTEL i7 CPU and 16G RAM. Among the ten workstations, one serves as the *Server*, the rest serve as the *Clients*. The system is deployed in a LAN, each with an independent IP address. The LTM is implemented as a MILP using C++ under Linux environment. The optimization tool used is CPLEX 12.1. The traffic data are extracted from the ASDI/ETMS which provides historical traffic records. The peak hours traffic on Mar 1, 2005 is used in which 2326 paths and 3054 flights were involved. Uncontrolled air traffic sets up the baseline for comparison purpose. The NAS-wide TFM optimization is executed with different Client/Server deployments. The correctness of the output will be verified first, then the computational efficiency will be shown.

The LTM is validated via output form the dual decomposition algorithm. Fig. 5 shows the value of objective function for the master problem which converges within 50 iterations. That means the whole optimization approaches the global optimum in an iterative manner. Fig. 6 shows the optimal air traffic flow of two sectors. As can be seen, the sector capacity constraint is well respected in optimized air traffic. In contrast, the uncontrolled air traffic exceeds the sector capacity sometimes during the peak hours. It is worth noting that the optimized air traffic has relatively balanced workload. Congestion in ZID74 is alleviated while traffic in ZNY75 is increased due to delayed aircraft. Clearly, delay is the tradeoff. In current LTM, delay is the only control to mitigate the congestion. If acceleration is allowed, the delay may be reduced. However, more sophisticated modeling and flight information are required to achieve such goal.

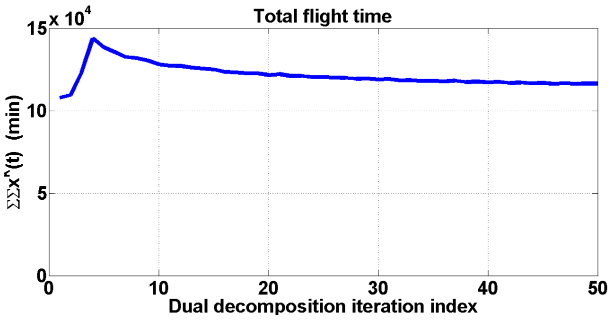
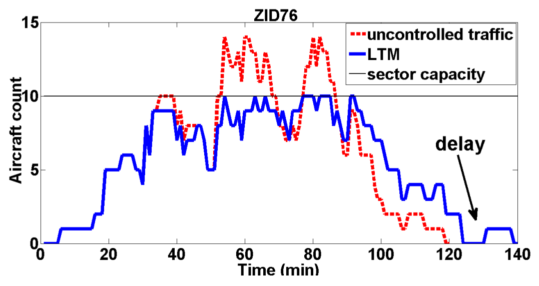
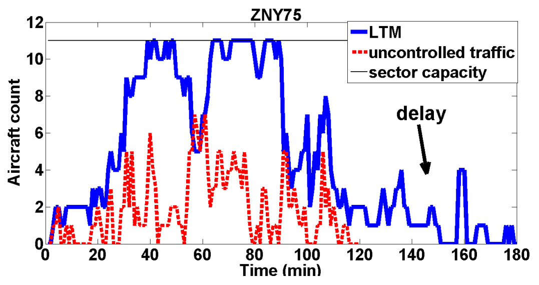


Fig. 5. Master problem objective converges

Fig. 7 presents the computational efficiency improvement by deploying different numbers of *Clients* and running different numbers of optimizer instances on each *Client*. In all cases, the program run 50 iterations to guarantee



(a)



(b)

Fig. 6. Optimized sector counts in two selected sectors

convergence. There are 2326 paths identified in the planning time horizon corresponding to 2326 subproblems. The lowest value of runtime is achieved by 9 clients each with 12 threads, which is around 6 minutes.

The monolithic version means that the *Server* and *Client* are running on the same workstation. Therefore the parallel computing is achieved simply by running multiple threads. As expected, the more optimizers are executed simultaneously, the more runtime is saved. But the saving is nonlinear. Initially, when the thread number is one or two, the usage of CPU is not maximized. Hence the increase of efficiency is notable. When the thread number nearly reaches the number of processors, the CPU is in full workload. Hence, the runtime reduction decreases. When the thread number exceeds the processor number, threads have to compete for CPU, then the computing power of a single machine reaches its bottleneck. For further gain of efficiency, multiple *Clients* must be deployed, as Fig. 7 indicates. Similarly, in y axle direction, the saving nonlinearly increases. Overhead for synchronization between the *Client* and the *Server* partially accounts for this observation. As client number increases, the *Server* has more *Clients* to respond. Another reason lies in the unbalanced workload. The *Server* evenly distributed the jobs to each *Client*, but the workload of each subproblem may differ a lot. A path with more links takes longer time for optimization. All *Clients* have to wait for the one with the heaviest workload at the end of each iteration. The more *Clients* is added, the more likely the workload is uneven. Thus more time is needed for synchronization. Ideally, the efficiency is maximized when the workload between *Clients* is homogeneous and the overhead of job distribution is commensurate with the computation of optimization. Even so, there should be a bottleneck for the parallel framework. From the trend shown in Fig. 7, it can be seen that the runtime reduction from 6 clients to 9 clients is relatively small. It can be predicted that it is impossible to reduce the runtime without limit by adding more clients. The lower bound for the runtime is the overhead.

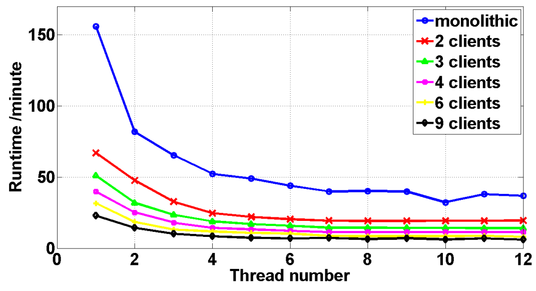


Fig. 7. Comparison of the runtime by different *Client* deployments

V. Conclusions and Future Work

This paper introduces a parallel computing platform used to improve the computational efficiency for NAS-wide TFM optimization. It boosts future NAS-wide ATM research in that the runtime of large-scale optimization problem is reduced from hours timescale to minutes timescale. Most importantly, the runtime reduction is independent of the complexity of the traffic network. The proposed framework can be easily adapted to the challenge of ever-changing air traffic problem.

References

[1] *JPDO Progress Report*, December, 2006.
[2] A.M. Bayen, P. Grieder., G. Meyer, C.J. Tomlin., "Lagrangian Delay Predictive Model for Sector-Based Air Traffic Flow," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 5, September-October 2005.
[3] P.K. Menon, G.D. Sweriduk, T. Lam, G. M. Diaz, and K.D. Bilimoria, "Computer-aided Eulerian air traffic flow modeling and predictive control," *AIAA Conference on Guidance, Navigation, and Control* , Providence, RI, August 2004, AIAA Paper 2004-2683.
[4] B. Sridhar, T. Soni, K. Sheth, and G.B. Chatterji, "An Aggregate Flow Model for Air Traffic Management," Journal. *Journal of Guidance, Navigation, Control. Dynamic*, pp.992-997, Jul-Aug. 2006.
[5] B. Sridhar, and P.K. Menon, "Comparison of Linear Dynamic Models for Air Traffic Flow Management," Proceeding. 16th *Int. Federal Automation, Control Conference (IFAC) World Congress*, Prague, Czech Republic, Jul.4-8, 2005.
[6] D. Sun, and A.M. Bayen, "Multicommodity Eulerian-Lagrangian Large-Capacity Cell Transmission Model for En Route Traffic," *Journal of Guidance, Control and Dynamics*. Vol. 31. No.3. May-Jun 2008.
[7] D. Sun, B. Sridhar, S. Grabbe, "Disaggregation Method for an Aggregate Traffic Flow Management Model," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 3, May-June 2010.
[8] Sun, D., Clinet, A., and Bayen, A.M., "A Dual Decomposition Method for Sector Capacity Constrained Traffic Flow Optimization," *Transportation Research-Part B*, 2010 (submitted for publication).
[9] Y. Cao, D. Sun, " Performance Comparison of Two Aggregate Air Traffic Models," *AIAA Conference on Guidance, Navigation and Control*, Toronto, Canada, August 2010.
[10] Rios, J., Ross, K., "Massively Parallel Dantzig-Wolfe Decomposition Applied to Traffic Flow Scheduling," *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, August 10-13, 2009.
[11] Rios, J., Ross, K., "Parallelization of the Traffic Flow Management Problem," *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 18-21, 2008.
[12] "Facility Operation and Administration," February 2010, Order JO 7210.3W, U.S. Department of Transportation, Federal Aviation Administration.