Development of a tool using file synchronization & parallel computing for EnergyPlus speedup

Vishal Garga , Nitin Dixita, Ravi Kumar Singha, Aviruch Bhatiaa, Rathish Sathyabama Arumugamb.

aCentre for IT in Building Science, International Institute of Information Technology, Hyderabad (India);

b Birla Institute of Technology and Sciences – Pilani, Hyderabad Campus, Hyderabad (India);

# Abstract:

This paper describes development of a tool - EPsync that uses file synchronization techniques and parallel computing over several nodes to speedup EnergyPlus group simulations and annual simulation. An annual simulation file is split into smaller chunks and distributed to various nodes using synchronisation tools. This tool can be used to increase the speed of group simulation files, parametric simulations and individual simulations. The EPsync tool can be used with any file synchronisation tools such as Dropbox, Google Drive, Sky Drive, Box etc. In our implementation we have developed the tool using Dropbox as it is available across all major platforms and utilises LAN synchronization. This tool is tested over Local area network (LAN) and Wide area network (WAN) and results are presented in this paper. For annual simulation speed gain up to 4.3 times is observed in comparison to run on single computer system.

**KEYWORDS:** EnergyPlus speedup; energy simulation; files synchronisation.

# Nomenclature:

|  |  |
| --- | --- |
| DOE | Department of Energy |
| Simulating Nodes | The nodes that run the simulation given by the master node. |
| EPsync | Name of the tool developed for File Synchronization |
| ESO | EnergyPlus Standard Output |
| IDF | Input Data File of EnergyPlus |
| Task | Task here refers to any task that can be broken down into smaller tasks, which can be run parallel. E.g. Energy Plus Simulation, rendering a video scene. |
| Overhead | Time taken for data transfer and collection between client and nodes over the network |
| Node | A system capable of running Energy Plus Simulation and which has its shared folder synced with all the other nodes in the network. |
| Network | The network of Nodes. The nodes can be over LAN or web in general. |
| Master Node | Any node that sends out the task. |
| Segment Simulation | Annual simulation is segmented into smaller multiple simulations called Segmented Simulations. Each segmented simulation can run on a dedicated CPU in a computer cluster or on different cores of the same computer |
| Segments | Input for each segmented simulation is referred to a segment |
| Speed Gain | Ratio of the run time of annual simulation to the time taken by the EPsync tool |

# Introduction

Building sector is one of the most energy consuming sectors in the world. To reduce the energy consumption in buildings various simulation tools available, which helps in analysing the energy performance of the building. EnergyPlus (Building Technologies Program: EnergyPlus Energy Simulation Software, 2012) is one such tool which was developed by the U.S Department of Energy. During the designing phase of a building, EnergyPlus allows the user to pre-compute the energy usage and thus optimize the building design. EnergyPlus performs a whole building energy simulation. EnergyPlus is a combination of the already existing tools like BLAST and DOE–2 and has an improved computational techniques and program structures (Crawley B. D. *et al.,* 2008).

Tianzhen *et al.* (2008) conducted a study to compare computer run time of building simulation programs. As per this study EnergyPlus v2.1.0 runs much slower than DOE-2 by an factor of 105 for a large office building to 196 for a hospital building for a 15 minutes time step and a factor of 25 for a large office building to 54 for a hospital building for a 60-minute time step. Though in the new versions of EnergyPlus (EnergyPlus v6.0), there have been significant reductions in the execution time in the order of 25-40% reduction in most simulations depending on specific features used (Building Technologies Program: EnergyPlus Energy Simulation Software: Archives, 2013), it is not sufficient to fix the gap between EnergyPlus and other energy simulation engines.

In a study conducted by Environmental Energy Technology Division of Lawrence Berkeley National Laboratory (LBNL), it is found that certain variable have significant impact on the simulation duration. Some simulation variables can have a very significant difference on the simulation duration of the building. These variables include the run period, Number\_of\_Timesteps\_per\_Hour for load calculations, heat balance solution algorithm, solar distribution and reflection calculation algorithm, system convergence limits, shadow calculation interval and the length of the warm up period (Tianzhen *et al.,* 2009).

To improve speed of energy simulations various techniques can be used. To increase the speed of multiple simulations EnergyPlus also the user to run the group simulation files in parallel. Up to 1024 simulation can be run in parallel in EnergyPlus, but this is limited by the hardware capabilities of the computer performing the simulation. The number of parallel simulation that could by run in a computer usually varies from 1 to 4 (Building Technologies Program: EnergyPlus Energy Simulation Software, 2012).

Though this technique reduces the simulation time of multiple files, simulation time for an individual file still remains large. Garg et al (2011) presented an approach uses data parallelization paradigm to increase the speed of single simulation run. A single simulation file was split into many smaller simulation files by changing the run period. These files were given to different computer to run these simulations in parallel. By splitting the annual simulation file into 12 monthly simulation files a speed gain of 3x to 6x was achieved.

Though this method achieved a very good speed, it could be used only on Linux based systems. The computers used for parallel simulation has to connected in cluster to run the parallel simulations.

In this paper file synchronization tools have been used to distribute the smaller simulation files to many computers. Using a file sharing synchronization tool provides an advantage to run parallel simulation across different platforms like Linux, Windows, and Macintosh etc. The parallel simulations now are no longer bound by a cluster and we can use computers located at different locations to perform the simulation.

A tool: EPsync has been built to demonstrate the use of file synchronization techniques. Dropbox has been used as the ﬁle synchronization service. The tool splits the given simulation file in small chunks and sends it to various computers present in the node for parallel simulation. It collects the results of the individual small simulations and presents the final results to the user.

# **2. Approach**

## 2.1 System Architecture

EPsync has been built to operate in a peer-to-peer network. EnergyPlus has to be installed in all the nodes in the peer-to-peer network in-order for EPsysnc to work. A peer-to-peer computer network is one in which each computer in the network can act as a client or server for the other computers in the network, allowing shared access to various resources. It means that all nodes are equipotent and there is no preference as such while selecting a node to perform a task. Most peer to peer networks are centralized to an extent that they have a dedicated central system to keep track of all the participating peers.

In this system, node availability is checked through the shared folder. Each node writes its status to the folder which is visible across all the nodes. This system is implemented as a peer to peer network. The nodes are classified into simulation node and master node.

Each simulation node consists of a five layer stack. These layers are as follows:

* Network Layer,
* Compression/Decompression Layer,
* Encryption/Decryption Layer,
* Application Layer,
* EP Layer.

Master node consists of a six layer stack. These layers are as follows:

* Network layer
* Compression/Decompression Layer,
* Encryption/Decryption Layer,
* Group Scheduler
* Application Layer,
* User Interface

These layers are independent of each other i.e. their execution does not depend on the functioning of other layers.

**Network Layer** is a medium through which data is transferred from one node to other. The network layer ensures that the copy of the shared folder is synced across all the nodes. The design of network layer depends upon the folder synchronization mechanism. In EPsync the network layer has been designed keeping Dropbox as a file synchronization mechanism.

**Compression/Decompression layer** compress/decompress the data which is received from the network layer. The compression helps in reducing the bandwidth usage. This data is sent to the subsequent layers for further processing.

**Encryption/Decryption layer** helps Encrypting/Decrypting the data. Though most of the file synchronization services implement their own encryption technique, but we add another layer of security to prevent any malfunctioning node to access unauthorised data. Encryption of data is optional to the user depending on the security level desired by the user. This will increase the processing time and reduces the speed gain.

**Application layer** takes care of the pre-processing activities and collating the output. The pre-processing tasks include splitting the simulation file into smaller components (task files), moving the task files to the correct location, calling the execution program.

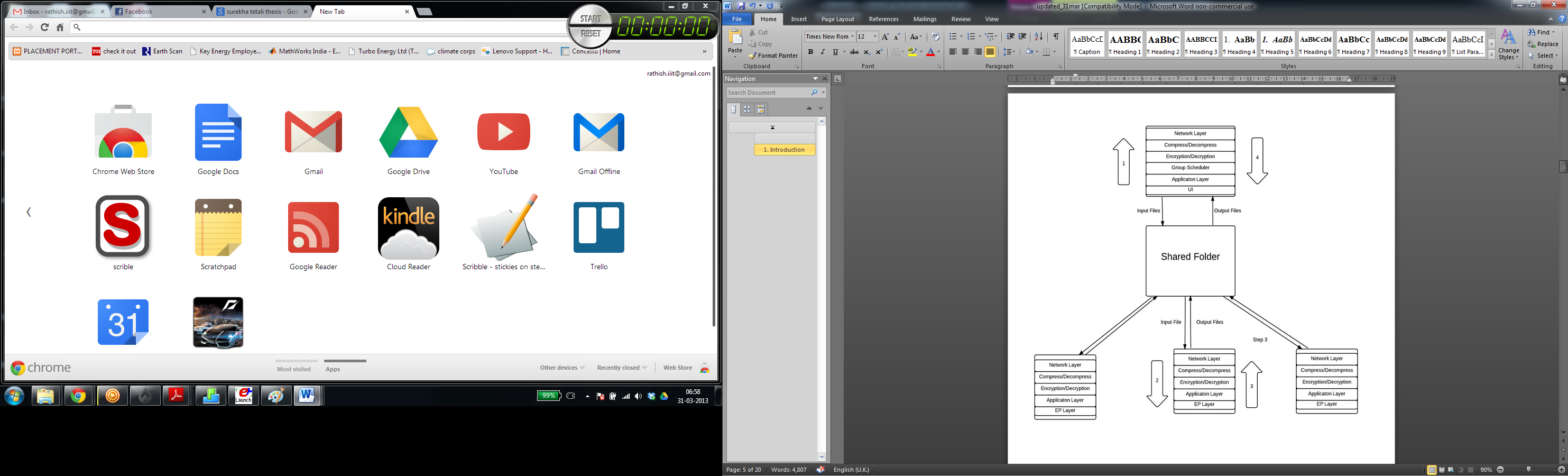


Figure 1: System Architecture for EPsync

**Execution layer** runs the given IDF file and gives the output back to the application layer. Building this as a separate layer ensures that in case of any updates in EnergyPlus we can upgrade EPsync tool to a higher version of EnergyPlus.

**Group scheduler** is a special layer available with the Master Node. It creates batches of task considering the priority and the size of the input files.

## 2.2 Nodes Identification & Status

When a system containing EPsync is connected in a peer-to-peer network for the first time, a file named nodelist.txt which is shared with the new system from the other system in the peer-to-peer network. If the system is found to be new on the network and unique identification number is assigned for the new system. This unique number is used as a reference for this particular system in the given network.

When the user runs EPsync in a computer in a given network, it has to know how many nodes are connected to the network and their status. A shared folder defined as ‘Status’ would be available on all nodes with EPsync. This shared folder contains a text file with a timestamp of the respective node. The timestamp is extracted from the file synchronization tool. Each node which is connected to the network keeps updating the timestamp in the respective text file every 5 minutes. All those nodes which have updated their timestamp in the last five 5 minutes are considered to be alive.

## 2.3 Data flow

The lists of live nodes are stored in a text file named AliveNodes.txt inside the shared folder. This file is also shared with all the nodes. Any node can act as the master node as there is no central server. Every node contains a local folder in which the IDF files will be stored after splitting. When the master node initiates the simulation for an IDF file, it runs the tool which splits the IDF file into smaller IDF files and stores it in a local folder. Then the list of alive nodes is generated. Once the input files are generated, the Group Scheduler decides which nodes perform the task. The nodes are selected based on the Node Selection Algorithm described in section 2.4. If the number of available nodes is less than the total number of simulations to be performed then the scheduler also decides the order of simulation. The scheduler decides the order based on the priority as specified by the user or by using the properties of the input file such as reporting variable frequency and size. The IDF files are renamed to the name of the respective node which is assigned to each file e.g. if a file has to be simulated by Node6, the file is renamed to Node6.idf.

Before moving the files to the shared folder, the files can be encrypted using any public key encryption technique. After encryption, the files are compressed to ensure minimal bandwidth usage. Since all the files being worked upon are simple text files, very high compression ratios can be achieved. All the IDF files are then moved inside a new task folder inside the shared folder. Every node keeps checking at regular intervals, if a new Task folder has been created.

After the node finds the task folder it check the file name in the task folder, for example, Node1 will look for a file named Node1.idf. If the node finds the respective file, it creates a new folder with the name of the node and moves the file inside the folder. It decompresses and decrypts the file.

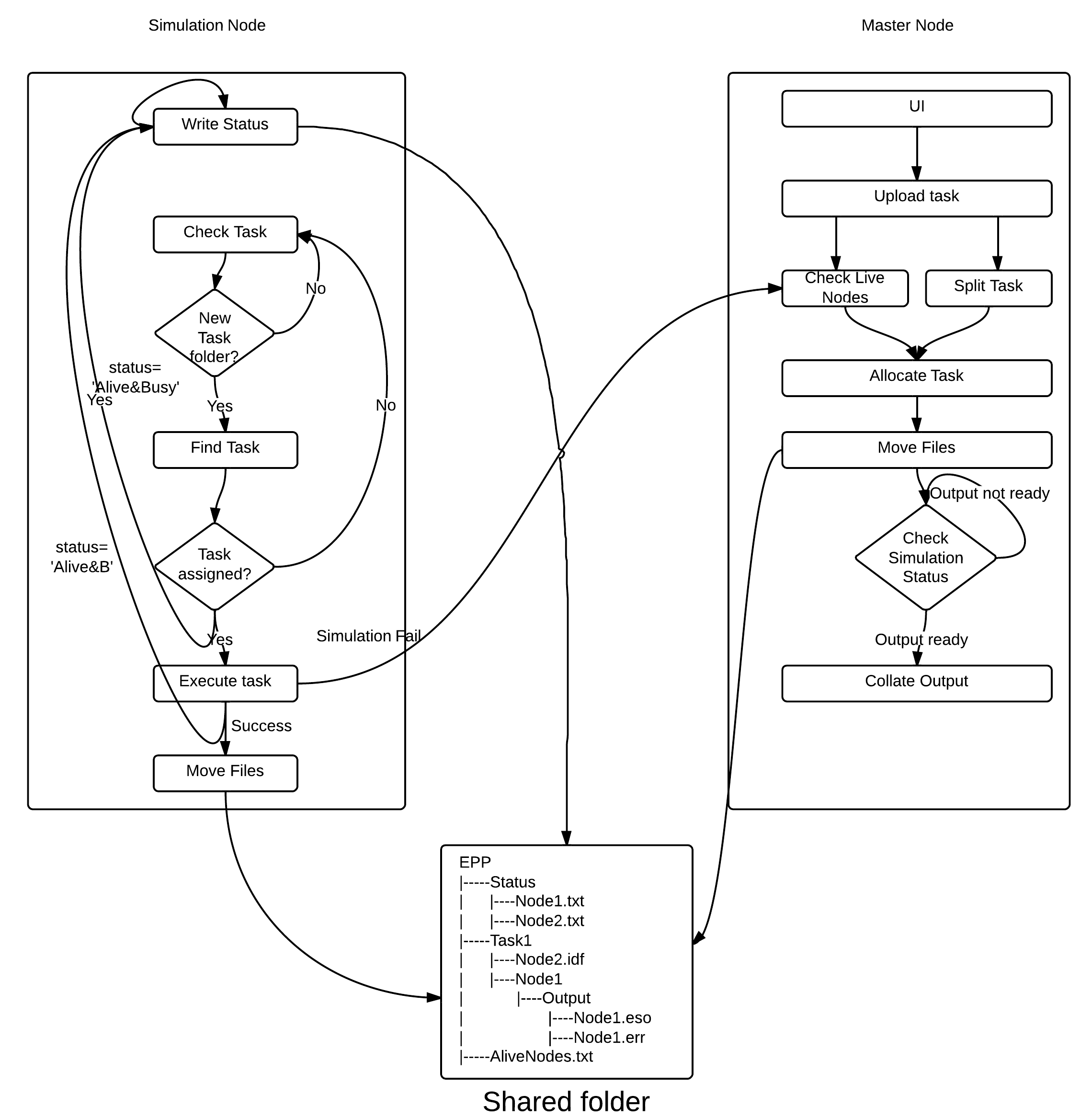


Figure 2: Data flow diagram for EPsync

The node then simulates the IDF file and the outputs are stored inside ‘Output’ folder inside the respective folder assigned to the node. The master node keeps checking if all the nodes have moved the output to the shared folder. Once all the nodes have simulated the IDF and moved the result to the shared folder, the tool takes all the outputs and collates them to generate a final output which is then returned to the user.

All the simulating nodes move their respective output folder to the shared folder. If all the shared folders are synced across all the nodes, each node has to download the output of all the other nodes. To avoid this, the 'Selective Sync' feature which is available with certain file synchronisation tools can be utilized. Selective sync facilitates syncing only the required sub folder and not the complete folder.

## 2.4 Node selection

The overall simulation time depends not only on the simulation time of IDF files but also on the overhead for synchronization of files. To optimize the total time taken for simulation, overhead time needs to be minimized. The overhead depends not only on internet speed and the bandwidth usage but also on which nodes in the network are selected to simulate the files. When a task is assigned, a group of nodes is selected which contains the required version of EnergyPlus. Out of the available nodes, required nodes are selected based on different parameters. These parameters for node selection can be divided into independent node parameters, parameters based on the relationship of the node with the master node and parameters based on input file.

Independent node parameters depend only on the performance of the node. Processor speed of the node will affect the simulation time of the system.

Another parameter is the percentage of times a node has successfully completed the simulation of the file assigned. If a particular node is not performing the task, the node is pushed down the priority list to increase the overall efficiency of the system. The nodes which are on the same network as that of the master node can communicate faster using features like LAN sync. With LAN sync, Dropbox will look for the new file on the LAN first, avoiding the internet usage to download the file from Dropbox servers, thus speeding up the syncing process.

Similarly for nodes which are not in the same network, preference order will depend on the round trip time for the ping request from the master node. To calculate the round trip time master node creates a dummy task folder and creates a separate file for each of the alive nodes. The master node issues a timestamp request through the files. Each node, upon detecting a new task folder, searches for a file with its name. Upon detecting a timestamp request, the nodes just write their current timestamp in the file. Round trip time is calculated based on the time it took for the change to be reflected on the master node.

The selection of node depends not only on the network and the individual nodes but also on the input file. An input file with less hourly variables indicates that processing speed is less of a bottleneck in considering the nodes for simulation. The size of the input file also affects the selection of nodes. A larger size input file indicates that internet speed can be a bottleneck in transferring the IDF files. So we give more preference to the nodes with smaller round trip time values.

## 2.5 Node Failure

In case of a node failure, where a node gets disconnected from the system, a new node has to be assigned the task in order to complete the simulation. If a node which is assigned a task gets disconnected before the simulation gets over, then the master node identifies that a particular IDF file has not been worked upon and the IDF file is re-assigned to some other node from ‘AliveNodes.txt’.

All the simulating nodes also keep rooting the current status of the EnergyPlus simulation to the master node through the shared folder. If the master node detects that simulation in a particular node is much slower than other nodes, then it can abandon that simulation and assign the task to a new node.

In case of master node failure, all the nodes need to select a node mutually and declare it the new master node. When a task is assigned, a text file 'CurrentNodes.txt' is created inside the task folder which contains the information of the nodes presently performing the task. The sequence in which node information is stored is also the priority order for master node selection in case of master node failure. 'CurrentNodes.txt' will have the master node listed first followed by all the nodes which are simulating the IDF files. Every node keeps checking if the node with just higher precedence is alive. In case the higher priority node disconnects from the network, it assigns the next higher priority node which is alive as its predecessor and keeps checking if that node is alive. If a particular node has no node left with higher priority which is alive, it declares itself as the master node. Thus a new master node will be assigned.

## 2.6 Shared Folder

The shared folder contains the following sub folders and files:

### 2.6.1 Status folder

Status folder is used to find the list of the nodes which are connected to the network. Each participating node has one file in the status folder. For example Node1 has a file named Node1.txt in this folder. Each node writes its current timestamp and status into its respective file every 5 minutes. The status of a node is 'Available' when it is free and 'Busy' when it is performing any simulation.

### 2.6.2 Task folder

It contains the IDF file assigned to the different nodes.The task folders are numbered as per the tasks created. Any Task folder initially contains the IDF files meant for different simulating nodes and a weather file. Each node detects its task and creates a new sub folder with its name. For example if there is a file named Node3.idf then Node3 would create a folder named Node3 and move the IDF file inside the folder. Once the execution is done, the node folder contains an Output folder containing all the output files.

### 2.6.3 AliveNodes.txt

This is a file which contains the list of alive nodes. This file is created each time a node hands out a task.

## 2.7 Adding new nodes to the system:

The system is dynamic in nature i.e. nodes can be added and removed very easily. If a node wants to get connected to the system, it needs to install the pre-requisite applications. The shared folder is then shared with the new node and a file is created inside Status folder with the name of the new node. Then once the node starts writing its status to the Status folder, the other nodes are able to discover its presence and the new node is treated similarly to all the other nodes of the system.

# 3. Use cases

## 3.1 Annual Simulation

A single annual simulation IDF file can be split into 12 monthly simulations as described in Garg Vishal et al (2011). These small simulation files can be shared in a peer-to-peer network to perform parallel simulation.

## 3.2 Parametric Simulation

Under parametric simulation, impact of the change in a few variables is studied on the overall energy consumption of the building. As such a number of simulations have to be performed with only minor changes in the IDF file as described by the varying parameter. Under this option, the user is allowed to enter up to two varying parameters and their range. A separate IDF file is created for each distinct value of the parameter. These IDF files are then sent to the Group Scheduler which schedules the simulation on different nodes.

## 3.3 Group Simulation

Under Group Simulation, the user provides a list of IDF files and a list of weather files. Each pair of IDF file and a weather file comprise a simulation unit. All these pairs are sent to the Group Scheduler which monitors the simulation.

# 4. Comparison between different File Sharing Tools:

EPsync works independently of the file synchronization technique. Any file synchronization service can be used to synchronize the shared folder across all the nodes. Some of the open source files synchronization services are Unilium, Cyberduck, Syncany, iFolder, owncloud, Duplicati and Seafile. Proprietary file synchronization services like Dropbox, Google Drive and Sky Drive generally have better performance both in terms of speed and security. Table 1 compares few of the available file synchronization techniques.

Table 1: Comparison of different file synchronization tools

|  |  |  |  |
| --- | --- | --- | --- |
| Features | Dropbox | Google Drive | SkyDrive |
| Free Storage | 2GB | 5GB | 7GB |
| Supported Platforms | Windows, Mac OS, Linux, iOS, Android, BlackBerry | Windows, Mac OS, iOS, Android | Windows, Mac OS, iOS, Android, Windows Phone |
| Features | Selective Folder Syncing, Events Tracking, Version History, Sharing Link, Facebook Group Integration | Selective Folder Syncing, Events Tracking, Version History, Sharing Permission Settings, Commenting on Files, Online Document Editor, Simultaneous Document Editing | Events Tracking, Version History, Sharing Permission Settings, Commenting on Files, Microsoft Office Web Apps (Word, PowerPoint, Excel, OneNote), Simultaneous Document Editing, Remote Access to Files on PC, OneNote Mobile App |
| Version History | 30 Days for All Files | 30 Days or 100 Document Revisions | 30 Days or 25 Document Revisions |
| File Upload Limit | Unlimited Through Desktop App, 300MB Through Website | 10GB Through Both Desktop App and Website | 2GB Through Desktop App, 300MB Through Website |
| Sharing | Simple Sharing Link | Customized Sharing and Access Settings | Customized Sharing and Access Settings |
| Security | 2-Step Verification, 4-Digit Passcode for Mobile App | 2-Step Verification Across All Google Services | Verification Code when Accessing PC Files Remotely |
| Supported File Types | Microsoft Office, Apple iWork, Audio, Video and Image Files | Adobe Illustrator (.AI) & Photoshop (.PSD), Autodesk AutoCad, Scalable Vector Graphics, Audio, Video and Image Files | Microsoft Office, Audio, Video and Image Files |

# 5. Tool description

Current version of EPsync is programmed in C. The file synchronization tool used is Dropbox. Dropbox can be used over different platforms including Linux. Dropbox also provides version history for all the files. It means that the individual nodes do not need to keep track of all the data they had worked on and it can be deleted from the shared folder once it is used to collate the final result but still if at a later point one needs to use the files, it can be recovered using the web application. This is not stored in the shared folder of the node but only on the Dropbox servers.

There is no upper limit size of the file that can be uploaded to the shared folder. Dropbox allows for events tracking under which we can track the syncing process and monitor the download and upload speed and the remaining time for syncing. Dropbox automatically throttles itself to 75% of the maximum upload speed to prevent any noticeable slowdown in browsing. Downloads are performed at the fastest download speed available.

Also LAN sync is available only on Dropbox. When a file is added to a computer's Dropbox folder, Dropbox initiates the syncing process as soon as it determines a change has been made to the file. All linked computers and shared folders will then download any new version of the file. With LAN syncing, Dropbox will look for the new file on Local Area Network first, bypassing the need to download the file from Dropbox servers, thus speeding up the syncing process considerably. It saves a huge amount of internet bandwidth when large numbers of participating nodes are connected over intranet.

# 6. Results

Five different annual simulation files have been simulated using EPsync in both WAN and LAN (using LANSync feature of Dropbox). The respective speed gains are presented in Table 2 and Table 3.

Table 2: Speed gain using EPsync in LAN networks

|  |  |  |  |
| --- | --- | --- | --- |
| Model Number | Time for simulation using single system (in seconds) | Time for simulation using EPsync (in seconds) | Speed Gain |
| Model 1 | 618 | 172 | 3.59 |
| Model 2 | 123 | 58 | 2.12 |
| Model 3 | 492 | 115 | 4.27 |
| Model 4 | 532 | 124 | 4.29 |
| Model 5 | 267 | 78 | 3.42 |

Table 3: Speed gain using EPsync WAN network

|  |  |  |  |
| --- | --- | --- | --- |
| Model Number | Time for simulation using single system (in seconds) | Time for simulation using EPsync(in seconds) | Speed Gain |
| Model 1 | 634 | 216 | 2.9 |
| Model 2 | 129 | 97 | 1.32 |
| Model 3 | 481 | 161 | 2.98 |
| Model 4 | 561 | 159 | 3.52 |
| Model 5 | 278 | 119 | 2.34 |

Table 4: Processing time of individual nodes in EPsync in LAN sync mode

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Model 1 | Model 2 | Model 3 | Model 4 | Model 5 |
| January | 115 | 18 | 67 | 83 | 41 |
| February | 102 | 16 | 71 | 85 | 37 |
| March | 118 | 18 | 68 | 95 | 39 |
| April | 140 | 19 | 67 | 87 | 44 |
| May | 122 | 18 | 68 | 85 | 39 |
| June | 108 | 24 | 82 | 88 | 42 |
| July | 100 | 19 | 7 | 89 | 40 |
| August | 102 | 19 | 69 | 87 | 38 |
| September | 100 | 22 | 71 | 92 | 39 |
| October | 136 | 23 | 71 | 89 | 41 |
| November | 93 | 15 | 68 | 88 | 39 |
| December | 102 | 17 | 72 | 90 | 38 |
| Slowest simulation | 140 | 24 | 82 | 95 | 44 |

Table 5: Processing time of individual nodes in EPsync in WAN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Model 1 | Model 2 | Model 3 | Model 4 | Model 5 |
| January | 118 | 17 | 65 | 79 | 46 |
| February | 108 | 18 | 68 | 81 | 40 |
| March | 115 | 23 | 68 | 89 | 39 |
| April | 143 | 21 | 72 | 87 | 45 |
| May | 123 | 18 | 68 | 84 | 41 |
| June | 112 | 20 | 78 | 88 | 42 |
| July | 104 | 19 | 75 | 91 | 39 |
| August | 103 | 20 | 71 | 87 | 38 |
| September | 99 | 18 | 71 | 93 | 39 |
| October | 136 | 22 | 68 | 87 | 43 |
| November | 93 | 17 | 70 | 90 | 40 |
| December | 104 | 17 | 72 | 89 | 41 |
| Slowest Simulation | 143 | 21 | 78 | 93 | 45 |

The time taken for EPsync to complete a simulation is the time taken by the slowest individual simulation plus the file transfer overheads of Dropbox. During any simulation run, file transfer overhead has to be added twice to account for the total time delay. This includes the delay for transfer all the IDF files from master node to other nodes and the results of all nodes to the master node. Thus, the total time of simulation using EPsync will be summation of total time for file transfer, time for file splitting/collation and slowest simulation time. It can be seen that EPsync improves the performance of the simulations, in terms of run time, by a factor of up to 4.3

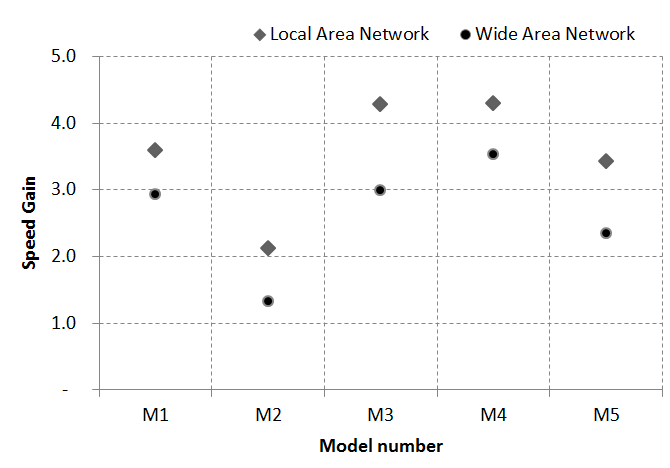


Figure 3: Speed gain for different models using LAN and WAN networks

. The speed gain for a larger file will be greater in general as the overhead of file transfer using Dropbox will be minimal as compared to the simulation time. As such this model is best suited for simulations where the run time is much higher than the file transfer overhead, which is typically in the range of 15-90 seconds. The testing was done both on LAN as well as on WAN to study the effect of LAN sync on the system. It can be concluded that LAN sync considerably improves the speed gain as compared to systems connected over WAN. The proposed approach can be very helpful in case of parametric analysis where a huge number of simulations are to be performed.

# 7. Conclusion

A concept has been developed and tested to speedup EnergyPlus using file synchronization techniques and parallel computing. A speedup of 4x has been achieved in the sample simulations. Using file synchronization enables us to use EnergyPlus parallel simulations in different platforms for the same simulation and it doesn’t require a dedicated cluster for distribution of the simulation files. The speed gain can further be increased by using file compression techniques. A reduction in file size of 80 – 90% in the simulation files were observed in the sample trails.

# 8. Acknowledgement

This work was partially supported by Joint Clean Energy Research and Development Center (JCERDC) for buildings called Center for Building Energy Research and Development (CBERD) funded by the Indian Ministry of Science & Technology, and U.S. Department of Energy and administered by Indo-US Science and Technology Forum in India.

# 8. References

“Building Technologies Program: EnergyPlus Energy Simulation Software.”, Accessed 10 January 2013, <http://apps1.eere.energy.gov/buildings/energyplus/>.

“Building Technologies Program: EnergyPlus Energy Simulation Software: Archives.”, Accessed 10 January 2012 <http://apps1.eere.energy.gov/buildings/energyplus/energyplus_archives.cfm#v6>

Crawley, Drury B, Jon W Hand, Michaël Kummert, and Brent T Griffith. 2008. “Contrasting the Capabilities of Building Energy Performance Simulation Programs.” *Building and Environment* 43 (4): 661–673. doi:10.1016/j.buildenv.2006.10.027.

Deru, Michael, Kristin Field, Daniel Studer, Kyle Benne, Brent Griffith, Paul Torcellini, Bing Liu, Mark Halverson, Dave Winiarski, Michael Rosenberg, MehryYazdanian, Joe Huang, and Drury Crawley. 2011. *U. S. Department of Energy Commercial Reference Building Models of the National Building Stock*. Technical Report NREL/TP-5500-46861, Golden, Colorado: NREL. [www.nrel.gov/docs/fy11osti/46861.pdf](http://www.nrel.gov/docs/fy11osti/46861.pdf).

“Dropbox”, Accessed 10 January 2012, [www.dropbox.com](http://www.dropbox.com).

Garg, Vishal, Kshitij Chandrasen, Jyotirmay Mathur, Surekha Tetali, and Akshey Jawa. 2011. “Development and Performance Evaluation of a Methodology, Based on Distributed Computing, for Speeding EnergyPlus Simulation.” *Journal of Building Performance Simulation* 4 (3): 257–270. doi:10.1080/19401493.2010.531142.

“Google Drive”, Accessed 10 Jan 2013, https://drive.google.com/.

Hong, Tianzhen, Fred Buhl, and Philip Haves. 2009. *EnergyPlus Run Time Analysis*. <http://escholarship.org/uc/item/36h4m5z0>.

Hong, Tianzhen, Buhl Fred, Haves Philip, Stephen Selkowitz, and Wetter Michael. 2008. *Comparing Computer Run Time of Building Simulation Programs*. <http://escholarship.org/uc/item/6504q6d0>.

LBNL, 2008. “GenOpt-generic Optimization Program.”, Accessed July 19, 2008, <http://simulationresearch.lbl.gov/GO/index.html>.

“Microsoft SkyDrive”, Accessed 10 January 2013, https://skydrive.live.com

Zhang, Yi. 2009. “‘ PARALLEL ’ EnergyPlus and the Development of a Parametric Analysis Tool.” In *Eleventh International IBPSA Conference*, 1382–1388. Glasgow, Scotland.

Anderson D. P., E. Korpela, and R. Walton, “High-performance task distribution for volunteer computing,” in E-SCIENCE ’05: Proceedings of the First International Conference on e-Science and Grid Computing