NT-SIM: A CO-SIMULATOR FOR NETWORKED SIGNAL PROCESSING APPLICATIONS

Stephen Won, Chung-Ching Shen, and Shuvra S. Bhattacharyya

University of Maryland
College Park, MD 20742 USA
{swon, ccshen, ssb}@umd.edu

ABSTRACT

In networked signal processing systems, network nodes that perform embedded processing on sensory inputs and other data interact across wired or wireless communication networks. In such applications, the processing on individual network nodes can be described in terms of dataflow graphs. However, to analyze the correctness and performance of these applications, designers must understand the interactions across these individual "node-level" dataflow graphs — as they communicate across the network — in addition to the characteristics of the individual graphs. In this paper, we develop a new simulation environment, called the *N*S-2 – *T*DIF *SIM*ulation environment (*NT-SIM*) — that provides integrated co-simulation of networked signal processing systems. NT-SIM systematically combines the network analysis capabilities provided by the Network Simulator (*ns*) with the scheduling capabilities of a dataflow-based framework, thereby providing novel features for more comprehensive simulation of networked signal processing systems.

Through a novel integration of tools for network and dataflow graph simulation, our NT-SIM environment allows comprehensive simulation and analysis of networked signal processing systems. We present a case study that concretely demonstrates the utility of NT-SIM in the context of a heterogeneous signal processing system design.

Index Terms— dataflow graphs, heterogeneous computing, co-simulation, scheduling

1. INTRODUCTION

Dataflow models are used to express the functionality of a variety of applications, including applications in many areas of signal processing (e.g., see [1]). In dataflow models of computation, applications are represented by directed graphs. Vertices (*actors*) of these graphs represent computational modules for running (*firing*) computational tasks, and edges represent first-in-first-out (FIFO) channels for storing data values (*tokens*), and establishing data dependencies between actors. When an actor is fired, tokens are consumed from and produced onto its input and output edges, respectively. Dataflow modeling of signal processing systems allows designers or design tools to schedule the firing of actors in ways that make efficient use of limited processing resources.

To utilize the benefits of dataflow graphs, the *targeted dataflow interchange format* (*TDIF*) development tool can be used to provide efficient mapping of application representations onto a variety of platforms [2]. TDIF extends the capabilities of the *dataflow interchange format* (*DIF*) [3] with dynamic dataflow software synthesis, cross-platform actor design, and dataflow-integrated features for application implementation. TDIF provides for dynamic dataflow modeling and scheduling to map signal processing applications onto heterogeneous platforms, and also provides retargetable actor construction, software synthesis, and instrumentation-based schedule evaluation and tuning [2].

However, dataflow-based modeling is typically not applied to networking aspects of networked signal processing applications. Network simulations involve link conditions and data protocols that are usually not represented using dataflow techniques. Network/application co-simulators address the issue of simulating the network conditions and the application at each node. However, most co-simulators today do not utilize dataflow-based modeling of the application (i.e., the intra-node functionality). As the range of network and distributed applications expands, it becomes increasingly important to develop methods to simulate the intra-node network conditions together with the dataflow models at the node level. Such a method would provide complete system analysis of networked signal processing applications without giving up the benefits of dataflow-based design practices at the level of individual nodes.

In this paper, we seek to bridge this gap by presenting the *N*S-2 – *T*DIF *SIM*ulation environment (*NT-SIM*). NT-SIM is a co-simulation tool that combines TDIF with the popular Network Simulator (ns-2) [4] to provide novel capabilities for experimentation with networked signal processing systems. NT-SIM is a flexible environment that allows designers to completely simulate systems at both the node and network levels. Dataflow-based design tools are available to assist in the development of layered sensing applications and other kinds of signal processing applications for which dataflow models can be applied to derive efficient placement and scheduling solutions. At the same time, ns-2 allows for detailed analysis of network properties and their effect on node information sharing. This allows designers to understand and validate the operation of network nodes as well as their interactions in the network.

We demonstrate that these objectives can be achieved through a case study of a sensor network performing image registration across multiple cameras with different views of an object. Through these experiments, we also demonstrate that our integration of network simulation with dataflow-based modeling and scheduling allows for new and useful analysis methods previously unavailable to designers in heterogeneous computing systems.

2. NT-SIM

Building on the capabilities of TDIF and ns-2, we have developed NT-SIM, a co-simulation environment that supports design and implementation of networked signal processing applications on heterogeneous platforms. NT-SIM allows simulation of end system behavior using TDIF, and network events using NSE.

Fig. 1 illustrates the execution order and interactions among components in the NT-SIM framework. Application behavior is specified based on dataflow modeling principles using the TDIF framework. To interface with the end system dataflow simulation and traffic generation for the network, the network behavior and protocols used by the nodes are defined by the OTcl script, and simulated by the NSE framework.

In NT-SIM, special dataflow actors called *interface actors* (*IAs*) are developed to allow the sending and receiving of information between NSE and TDIF. In contrast to conventional dataflow actors, which represent functional components from the application specification, IAs are responsible for traffic generation from TDIF-based modeling subsystems, and injection of this generated traffic into the NSE framework. IAs are also responsible for time synchronization between the cooperating TDIF- and NSE-based simulation environments. This collection of IAs in a TDIF-based dataflow subsystem makes the subsystem appear as a single node within an enclosing ns-2 network topology.

The architecture of NT-SIM is designed to preserve the dataflow principles provided by the TDIF environment throughout all TDIF-based subsystems, including the interactions that occur at the interfaces of these subsystems (i.e., at the IAs). The designer is responsible for specifying the distribution of actors to the nodes in the network graph. In the NT-SIM framework, the designer develops the system in a hierarchical manner: actor design using TDIF, dataflow graph design at each network node using DIF, and network graph design using ns-2. The FIFO communication channels in DIF act as bridges between actors in the dataflow graph. Correspondingly, the IAs act as bridges between dataflow graphs that are placed on different network nodes. In NT-SIM, dataflow subsystems can be suspended (e.g., as they wait for data) and resumed arbitrary numbers of times while the overall network is being simulated, thus allowing for simulation of complex and tightly-coupled feedback behaviors across the network.

Thus, NT-SIM provides designers with a hierarchical, modular process for modeling and experimenting with networked signal processing systems. NT-SIM also provides a useful target for incorporating additional levels of automation in the design and simulation processes. For example, protocol configurations and associated implementation details can be determined and optimized automatically by incorporating associated IA synthesis capabilities within the TDIF synthesis engine. Building on NT-SIM to develop such new automation and optimization capabilities is an interesting and useful direction for future work.

The processes of design and experimentation using NT-SIM are demonstrated more concretely in Section 3, which provides a case study involving the development of a visual sensor network.

3. CASE STUDY: VISUAL SENSOR NETWORK

We demonstrate the utility of NT-SIM with a case study of simulating a visual sensor network designed to perform image registration on different views of the same object. This case study is motivated by the rapidly developing field of distributed sensing and its application in tasks such as layered sensing, surveillance, and videoconferencing [5, 6].

Instead of gaining knowledge about the environment through a small number of expensive cameras, multiple low-cost cameras can be utilized to provide more complete pictures for challenging, high-level vision tasks such as image registration or tracking [7]. This requires the cameras to be networked together, and to perform collaboration tasks among themselves to optimize key metrics, such as real-time performance, power consumption, and image processing accuracy. Such metrics generally depend on node-network interactions, and thus conventional simulation methods, which consider only network and node characteristics in isolation, are not sufficient. NT-SIM is able to assist in the design of such distributed sensing systems by providing the designer with integrated capabilities to simulate algorithms and applications at the network and node levels.

3.1. Distributed Vision Sensor Systems

Visual sensor networks (VSNs) are comprised of groups of networked visual sensors with image capture, computation, and wireless communication capabilities. To maximize the effectiveness of a VSN, collaboration among the sensors can take place with the exchange or fusing of visual information from similar or different perspectives of an area [7]. This

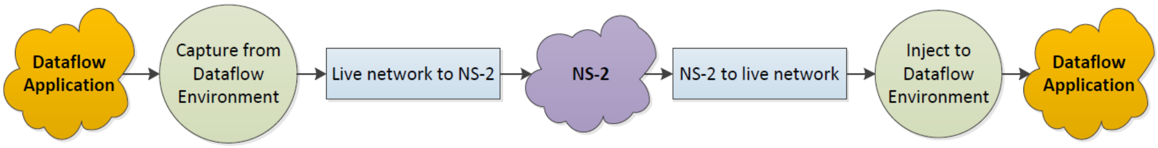

Fig. 1. Illustration of the interaction between dataflow applications and network simulations in NT-SIM.

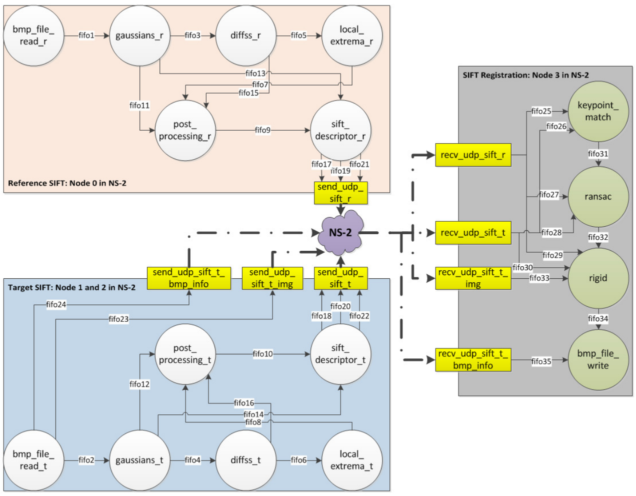

Fig. 2. A dataflow graph model of SIFT-based feature detection and image registration across a network.

allows the information to be used in tracking, panoramas, and registration.

The scale-invariant feature transform (SIFT) [8] is an algorithm that can be used to fuse together images from multiple cameras that are observing the same object. SIFT uses the difference of Gaussian (DoG) to detect feature keypoints at different visual scales. To highlight strong features in the images, the eigenvalues of the Hessian matrix of the image are used to highlight reliable features to use. Results can improve with random sample consensus (RANSAC), which removes outliers and erroneous features detected by the algorithm. Fig. 2 shows a dataflow graph model of the SIFT algorithm. Here, the SIFT algorithm is used to register two images with different views of the same object.

Each sensor node in a VSN has to fulfill application requirements while running under constraints involving memory, performance, data rates, and energy [9]. By distributing actors appropriately across the network, more processing-intensive tasks can be performed at one or more stationary systems that are connected to power sources, while simpler tasks are handled by the sensor nodes. This allows energy on the sensor nodes to be conserved while the computationally-intensive task of image registration is carried out, and also helps to improve the performance of image registration by allowing use of more powerful (less power constrained) platforms for the registration tasks.

```
module CUDA sift_descriptor_r

output output1 sift_token
output output2 sift_token
output output3 sift_token

input input1 oframes
input input2 gss

mode init
mode exe
```

Fig. 3. TDIF code for the SIFT descriptor actor.

In our case study on a SIFT VSN, we experiment with this approach of heterogeneous computing and distribution-based optimization of energy and performance for the SIFT application in a VSN. This experimentation is carried out through mapping of the dataflow graphs for distributed signal processing onto separate network nodes, configuration of IAs in TDIF for appropriate communication among the nodes, and simulation using NT-SIM.

3.2. Actor Design

Each of the actors in the SIFT algorithm is modeled using the TDIF environment. For this purpose, the SIFT algorithm is broken into smaller procedural units to be modeled with actors. At this level of NT-SIM, the actors are not assigned to any particular nodes in a network. The focus at the actor design level of NT-SIM is to create actors that are represented by the TDIF language. In this phase of the design process, designers specify the target language of each actor, along with the inputs, outputs, required parameters, and possible execution modes for the actor. Fig. 3 shows the TDIF file for the SIFT descriptor actor, which passes the SIFT descriptor to the keypoint matching, RANSAC, and rigid transformation actors. The SIFT descriptor actor represented in Fig. 3 is specified as a CUDA-targeted actor for GPU-based implementation.

As an example, Fig. 4 shows TDIF code for sending an image from the actor representing the capture of the target image to the network simulated by ns-2. For simplicity and clarity in the illustration, we design the network to follow the UDP protocol. As a result, the image-sending actor represented by Fig. 4 takes in the address and port number as character-string parameters, and these parameters are em-

```
module C send_udp_sift_t_img

input input_image image_token*

param send_addr char*
param send_port char*

mode init
mode send
```

Fig. 4. TDIF code for sending an image to NSE via the UDP protocol.

ployed by the actor in addition to any inputs coming from other actors in the enclosing dataflow graph subsystem.

3.3. Actor Separation at the Node Level

In NT-SIM, the application that runs on each network node is represented by a specification in the DIF language. To optimize the energy and performance of the SIFT VSN, actors are split onto different network nodes depending on their roles in the overall application graph. In this case study, actors are distributed across network nodes depending on whether they perform feature detection or image registration. This results in multiple dataflow graph subsystems with each subsystem corresponding to a single network node. Each of these subsystems can be specified using a DIF file that defines the actors as vertices and the connections between them as edges in the associated dataflow graph.

The current version of NT-SIM systematically integrates designer-provided tests and schedules into the overall network simulation, and automates the execution of this simulation across the entire network. Thus, NT-SIM bridges the gap between network- and dataflow-graph-level simulation in networked signal processing systems, and provides novel capabilities into which existing and newly developed dataflow scheduling techniques can be integrated to further enhance simulation automation and design space exploration.

3.4. Network Creation

When using NT-SIM, the designer creates a Tcl script that models the network topology on NSE to simulate the network. In order to use NSE on ns-2, the `RealTime` scheduler has to be used with the simulator. Nodes are declared along with the network objects and agents. When using the UDP protocol, each of the network objects has to declare the IP address and port number in the script. These network objects are attached to their corresponding agents. Afterwards, the connections between nodes can be defined, along with the bandwidth, delay, and queue behavior for each connection.

Each agent is attached to a node. If the nodes share a common link, then the agents are also connected. Afterwards, NSE can be run. Fig. 5 illustrates the network topology used in our SIFT VSN case study.

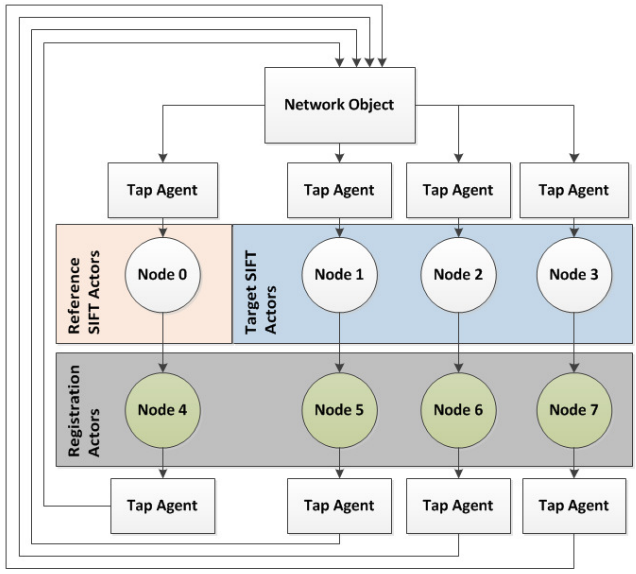

Fig. 5. The topology represented by the Tcl script for the SIFT sensor network.

3.5. Simulation of the Distributed System

After the actors, dataflow graph subsystems (the portions of the dataflow graph that are mapped onto individual network nodes), and the network have been specified, the overall system can be simulated using NT-SIM. The Tcl script for the network is run using NSE. This allows network connections to be made between the TDIF and ns-2 environments. Separate test and DIF files are required for each VSN node. After the executables have been generated for each VSN node, they can be run — concurrently with simulation of the resulting network traffic — to send and receive data to and from NSE, respectively.

The SIFT sensor network is simulated on a 3GHz PC with two Intel Xeon CPUs, 3GB RAM, and an NVIDIA GTX260 GPU. The `gcc version 3.4.4` and `nvcc version 3.2` compilers are used in the back end of the implementation process.

The functional accuracy of NT-SIM was verified through simulation of the SIFT VSN case study. End systems (network nodes) representing reference and target image sensors that can perform feature detection were supplied with only the reference and target image shown in Fig. 6. Functional accuracy was validated by the match between the produced, registered image and a ground-truth, registered image provided by the simulation of the single-node SIFT algorithm.

Fig. 6. (Clockwise from top left) Reference image, target image, and registered image from the simulated SIFT VSN network.

4. CONCLUSIONS

We have introduced NT-SIM as a co-simulation tool that combines the dataflow paradigms of TDIF and DIF for actor and dataflow graph design, respectively, and the network simulation capabilities of NSE. The resulting tool provides useful new capabilities for integrated simulation of networked signal processing systems. We have demonstrated that using NT-SIM, a designer can simulate a complete, networked system comprised of a distinct application subsystem on each network node with actors modeled using formal dataflow-based representations. The useful features of NT-SIM include its modular design flow, where actors are designed using the TDIF tool, application graphs are modeled in the DIF framework, and the network is represented in ns-2.

Useful directions for further development of NT-SIM include automating the partitioning of an application dataflow graph across a network through the TDIF synthesis engine; application of instrumentation actors in TDIF to encapsulate relevant network performance measurements provided by NSE; incorporating different network protocols along with reuse of the associated protocol code as TDIF actor library components; and exploration involving other kinds of networked signal processing applications, such as distributed speech processing [10] and adaptive stream mining [11].

5. REFERENCES

[1] S. S. Bhattacharyya, E. Deprettere, R. Leupers, and J. Takala, Eds., *Handbook of Signal Processing Systems*. Springer, 2010.

[2] C. Shen, H. Wu, N. Sane, W. Plishker, and S. S. Bhattacharyya, "A design tool for efficient mapping of multimedia applications onto heterogeneous platforms," in *Proceedings of the IEEE International Conference on Multimedia and Expo*, Barcelona, Spain, July 2011, 6 pages in online proceedings.

[3] C. Hsu, M. Ko, and S. S. Bhattacharyya, "Software synthesis from the dataflow interchange format," in *Proceedings of the International Workshop on Software and Compilers for Embedded Systems*, Dallas, Texas, September 2005, pp. 37–49.

[4] K. Fall and K. Varadhan, *The ns Manual (formerly ns Notes and Documentation)*, November 2011.

[5] M. Bryant, P. Johnson, B. M. Kent, M. Nowak, and S. Rogers, "Layered sensing: Its definition, attributes, and guiding principles for AFRL strategic technology development," Sensors Directorate, U.S. Air Force Research Laboratory, Tech. Rep., May 2008.

[6] A. C. Sankaranarayanan, A. Veeraraghavan, and R. Chellappa, "Object detection, tracking and recognition for multiple smart cameras," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1606–1624, October 2008.

[7] Y. Bai and H. Qi, "Feature-based image comparison for semantic neighbor selection in resource-constrained visual sensor networks," *EURASIP Journal on Image and Video Processing*, 2010, doi:10.1155/2010/469563.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "Wireless multimedia sensor networks: Applications and testbeds," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1588–1605, October 2008.

[10] C. Shen, W. L. Plishker, D. Ko, S. S. Bhattacharyya, and N. Goldsman, "Energy-driven distribution of signal processing applications across wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 6, no. 3, June 2010, article No. 24, 32 pages.

[11] R. Ducasse, D. Turaga, and M. van der Schaar, "Adaptive topologic optimization for large-scale stream mining," *IEEE Journal on Selected Topics in Signal Processing*, vol. 4, no. 3, pp. 620–636, June 2010.