A Distributed Computing Framework for Real-time Detection of Stress and of its Propagation in a Team

Parul Pandey, *Student Member, IEEE*, Eun Kyung Lee, *Student Member, IEEE*, and Dario Pompili, *Senior Member, IEEE*

***Abstract*—Stress is one of the key factors that impacts the quality of our daily life: from productivity and efficiency in production processes to the ability of (civilian and military) individuals in making rational decisions. Also, stress can propagate from one individual to other working in close proximity or towards a common goal, e.g., in a military operation or workforce. Real-time assessment of the stress of individuals alone is, however, not sufficient as understanding its *source* and *direction* in which it propagates in a group of people is equally – if not more – important. A continuous *near real-time in-situ* personal stress monitoring system to quantify level of stress of individuals and its direction of propagation in a team is envisioned. However, stress monitoring of an individual via his/her mobile device may not always be possible for extended periods of time due to limited battery capacity of these devices. To overcome this challenge a novel distributed mobile computing framework is proposed to organize the resources in the vicinity and form a mobile device cloud that enables *offloading* of computation tasks in stress detection algorithm from resource constrained devices (low residual battery, limited CPU cycles) to resource rich devices. Our framework also supports computing parallelization and workflows defining how data and tasks are divided/assigned among the entities of the framework are designed. The direction of propagation and magnitude of influence of stress in a group of individuals are studied by applying real-time, in-situ analysis of *Granger Causality*. Tangible benefits (in terms of energy expenditure and execution time) of the proposed framework in comparison to a centralized framework are presented via thorough simulations and real experiments.**

***Index Terms*—distributed mobile computing; real-time stress detection; Granger causality; experiments.**

I. Introduction

Motivation: Stress is one of the key factors affecting physical and mental wellbeing, which is essential in any environment where high level of performance is required and sought. Research has shown that doctors and army personnel experience high level of stress as tolerance for errors in their profession is very low [7]. High-stress can significantly impair the ability to perform tasks and to make rational decisions, which may impact a patient's life in case of doctors and national security in case of army personnel [21].

Vision: Real-time stress detection provides knowledge of the current level of stress experienced by a person and motivates him to take actions aimed at enhancing his current productivity. Stress can also *propagate* from one individual to others working in close proximity or towards a common goal, e.g., in a military operation or workplace. The members in the team can be affected by *anxiogenic behavior*, i.e., a behavior that induces anxiety and stress, of any/few members of the team on other members. As an example, consider the scenario (civilian or military) in which a team has to be formed for a task: in this case, real-time stress detection can give us insights into the current stress levels of different members of the team. Also, the employees/soldiers who can perform tasks without being overwhelmed by those at higher ranks, or by the anxiogenic behavior of other team members, should be selected. This capability will empower people who are in better condition *in the field* (e.g., less stressed) by putting them in charge of the situation and providing medical help to people who have high level of stress or induce stress on others. This will improve effectiveness and efficiency as well as allow dynamic hierarchy reorganization beyond existing ranks and roles. Hence, our interest is in real-time *quantification* of stress of an individual and in understanding the *direction* as well *magnitude* (i.e., extent) of influence of stress from one person to another in the same group.

Our approach: We envision a continuous near real-time, in-situ personal stress monitoring system to quantify the level of individual stress and its direction of propagation in a group of people. The acquisition of physiological signals is done via non-invasive sensors and are sent wirelessly to the personal mobile devices of the group members. The quantification of stress is done on the mobile device of each person. However, stress detection for extended periods of time can lead to significant energy expenditure of these mobile devices. These problems are of significance as current mobile devices have limited battery capacity, CPU cycles, and memory size. Considering the performance trends in mobile-device architecture and battery capacity so far, it is unlikely that we will see any major improvements in their capabilities in the near future [15]. In order to overcome the limitations of individual computing devices and to enable in-situ, near real-time detection of stress and its direction of propagation in a team, we propose to *offload* the computation tasks from resource limited mobile devices to multiple static and mobile devices (such as smart phones, tablets, netbooks, and laptops) in the vicinity. Some work has been initiated whereby the storage of medical data and computation of health monitoring applications have been moved to powerful and centralized remote computing platforms such as the Cloud [14], [19]. However, execution in the Cloud may not always be possible

The authors are with the Dept. of Electrical and Computer Engineering, Rutgers University–New Brunswick, NJ, USA.
E-mails:{parul_pandey, eunkyung_lee, pompili}@cac.rutgers.edu
A preliminary version of this work appeared in the Proc. of the ACM International Conference on Body Area Networks (BodyNets), Boston, MA, Sept. 2013 [13].

due to (i) frequent disconnections, (ii) low data rate, or (iii) inability to access a wireless connection.

To enable the detection of the source of stress and its direction of propagation in time, we propose to use *Granger Causality*. In neuroscience G-Causality has established itself as one of the promising approaches to reveal the direction of influence between brain areas by analyzing temporal precedence: *if a signal change in area A consistently precedes a signal change in area B, then A is said to Granger-cause B [16]*. In this work we extend Granger-causality approach and advocate the use of this tool to study causality between physiological signals from two different individuals.

Challenges: Real-time quantification of the magnitude of stress and of its direction of propagation via our framework faces both communication and computation challenges. Firstly, large amount of data has to be moved from the sensor nodes, which are attached to members of the group, to the nearby computing nodes (*communication bottleneck*). Secondly, estimation of stress and G-Causality of individuals in a group is a data-intensive task that requires solving multiple linear regression problems (*computation bottleneck*); also, the computation complexity of such pairwise analysis increases *linearly* for quantification of stress and *quadratically* for G-Causality with the number of people in the group.

Contribution: To address these challenges, we propose to quantify stress and its propagation (via G-Causality) of a team in a distributed manner by parallelizing the execution of computing tasks. We refer to our earlier work on mobile computing grid to enable distributed computation [26]. The distributed grid is divided into two major entities, namely: 1) *middleware*, which solves the communication bottleneck by moving the data within the network efficiently, and 2) *framework*, which takes care of the high computation aspect of the problem by assigning computation tasks "optimally" to the different entities of the distributed grid (computation nodes). Figure 1 envisions two scenarios, a civilian (left) and a military (right), where sensors on the body of medical personnel or soldiers, respectively, are used to collect continuously and non-invasively various physiological signals (such as Galvanic Skin Response (GSR), Electrocardiogram (ECG), body temperature, and respiratory data among others) and nearby rugged computing devices – organized by our mobile computing framework – are used for real-time compute-intensive group-stress analysis.

The major contributions of this work are as follows.

- Enabling estimation of stress of a group of individuals in real time through the collaboration of local computing resources so to form a *distributed mobile grid.*
- Enabling compute-intensive group Granger-Causality analysis of stress using our mobile-grid framework aimed at estimating the direction of stress propagation.
- Demonstration via simulations and real experiments of the benefits of our distributed computing framework.

Paper outline: The rest of the article is organized as follows. In Sect. II we discuss the state of the art in the area of real-time stress detection; in Sect. III, we present our proposed solution, which is aimed at enabling real-time quantification of stress of a group of people and at estimating the direction of stress propagation via distributed G-Causality; in Sect. IV, we describe our experimental methodology, discuss the benefits of our distributed computing framework, and present simulation and experimental results to demonstrate the potential of G-Causality in estimating propagation of stress; finally, in Sect. V, we present our conclusions and plans for future work.

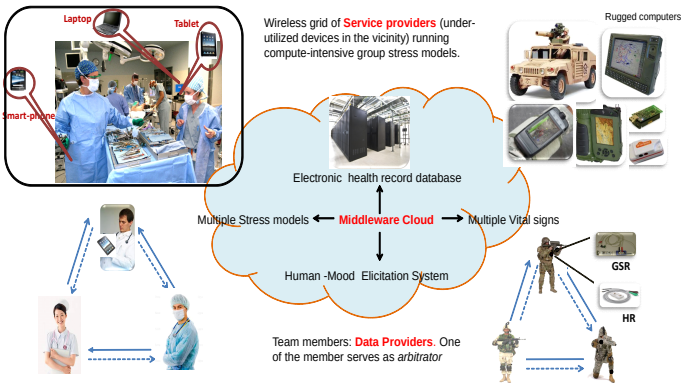


Fig. 1: Distributed resource provisioning framework for real-time monitoring of stress and of its propagation in a team.

II. Related Work

Different physiological signals are considered in the literature for real-time stress detection, e.g., GSR, finger temperature, blood volume pulse, pupil dilation, and eyetracking [4], [28]. The main characteristic of these works lies in the fact that signals are acquired in a non-intrusive manner; furthermore, these physiological signals provide a predictable relation with stress variation. Once the physiological measurements have been collected, a variety of techniques such as support vector machines, fuzzy logic, and Fisher linear discriminant analysis are used for stress estimation [8], [23], [27]. Work has also been done to estimate stress detection of drivers [12], [11] in real time to improve their mental state by managing music selection in the vehicle and by providing various distraction management techniques. In [9], the authors propose using virtual reality to expose a person to various potential stressors and design a decision support system based on machine learning to estimate the person's stress levels. Effective coping skills are given to the person based on their stress levels to prepare him/her for a similar situation. In [5] authors use only GSR sensor data to detect changes in stress level of a person. Their main task is to look for changes in the observed GSR data from a not stressed state to an aroused state (stressed state). The authors also present techniques on how to remove noise and disturbances in the GSR signal. In [20] authors present SVM based models for stress detection form physiological measurements such as ECG, GSR, respiratory features, and body temperature. In [22], [11] the authors enable stress detection by including activity data from accelerometer sensors. This analysis takes into account the physical activity being performed by the user while estimating stress to distinguish between physical and mental stress.

The work in the area of real-time stress detection has so far focused on stress detection of an individual only. Furthermore,

current works do not study the direction of propagation of stress in a group to determine the source of stress so to improve group dynamics. Also, all the work done so far suffers from the curse of extreme centralization. i.e., stress estimation is done at a centralized location, hence, creating a single point of failure. Our framework, on the other hand, enables group stress detection by offloading tasks to those nearby resources that are willing/able to offer their computational capability. It also overcomes the issues faced by failure of a single computation node by having a resource pool to distribute tasks to in case of failure of one node. Our middleware provides the required communication infrastructure to distribute tasks and receive results from nearby entities under *uncertain* network conditions and device availability.

III. Proposed Solution

We present here our solution to enable real-time stress detection and its direction of propagation for a group of people. We begin by giving details of a representative scenario in which stress of a group of people is monitored. We present our experimental setup to collect physiological measurements from non-invasive sensors. Next, we explain our proposed methodology for real-time group stress estimation. Our methodology consists of two sequential time periods, namely, *acquisition period*, to collect physiological data from group members and *stress detection period*, to determine stress levels of each member of the group as well as the source of stress in the group via G-Causality. Next, we describe the entities of our proposed distributed computing framework, which provides the computation and communication infrastructure to execute stress-detection algorithms in a distributed manner. We also show how the magnitude and direction of propagation of stress can be determined for a group of people by exploiting G-Causality. At the end of this section, we discuss how distributed computation of G-Causality is performed using our framework to monitor propagation of stress in a group.

A. Methodology for Continuous Stress Detection

Representative scenario: We consider a scenario where a group of people require constant monitoring of physiological signals so to profile their stress levels and prevent any risk of fatal-health condition. This scenario can be seen in a military base or workplace where high levels of stress can impact decision making, or in a hospital or assisted-living facility where constant monitoring of vital signs can help detect any condition that requires immediate medical attention.

Data Acquisition: Among the various physiological signals such as Galvanic Skin Response (GSR), Electrocardiogram (ECG), Electromyogram (EMG), body temperature, and respiration, the authors in [12] have shown that skin conductivity metrics (derived from GSR) and heart rate metrics (derived from ECG) are the most closely related to stress level of an individual, hence, we use these two physiological signals for stress detection. To detect stress in the people being monitored, we measure physiological signals (here GSR and ECG) via non-invasive sensors attached to their body such that they do not interfere with their daily activities. To enable continuous

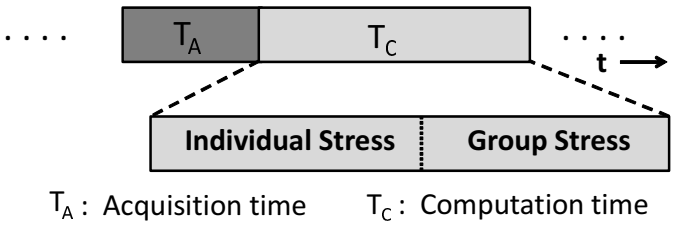


Fig. 2: Proposed acquisition and computation methodology for real-time individual and group stress detection.

real-time detection of stress of individuals *in a group*, we propose a novel methodology, shown in Fig. 2. Real-time stress detection is executed over two sequential time periods, namely, *acquisition* (T_A) and *computation* (T_C) periods. Acquisition period involves collection of data via non-invasive sensors and the computation period we first quantify the stress experienced by people in the team followed by estimating the direction of propagation of stress among team members. We briefly explain the two sequential time periods.

(a) (b)

Fig. 3: SHIMMER sensors (a) Electrocardiogram (ECG) and (b) Galvanic Skin Response (GSR).

The acquisition period (T_A) involves collecting data from all the members in a group for whom the stress has to be monitored. The GSR sensors measure the change in electrodermal activity (increase in conductance) as sweat glands are stimulated for a hydrate solution. An ECG signal is a recorded tracing of the electrical activity generated by the heart. We derive various metrics from these signals to quantify the stress experienced by a person. In our experiments we use non-invasive GSR and ECG sensors provided by Shimmer Sensing [1]. Figure 3(a) and (b) show the ECG and GSR sensors, respectively. These sensors communicate data to the smartphones via Bluetooth. The sampling rate of the GSR sensors is set to 50 Hz and to 100 Hz for the ECG sensors. If the application requires differentiation between mental and physical stress, then the activity data collected via accelerometer or annotations from the user (information provided by the user that s/he is experiencing high physical stress, e.g., exercise or climbing stairs) can be used to detect the time period of the physical activity. If the focus is only on mental stress, then the application can choose to not consider data acquired during the time period that is annotated as high physical activity.

TABLE I: **Different phases of experiment to quantify stress.**

Phases	Types of Questions	Num. of Questions	Time [min]
Phase-1 Relaxed Stage (RX1)			2.5
Phase-2 (STR1) Difficulty Level: Low	Stroop Test Logical Questions Arithmetic Questions	1 3 3	2 4 4
Phase-3 (STR2) Difficulty Level: High	Memory Based Stroop Test Logical Questions Arithmetic Questions	1 1 2 2	1 1 2.5 2.5
Phase-4 Relaxed Stage (RX2)			3

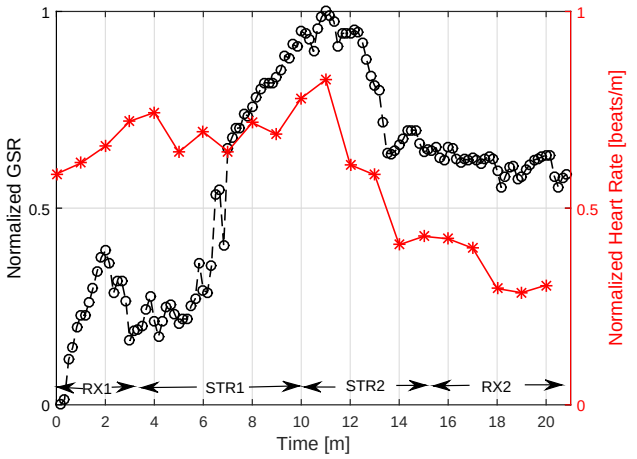


Fig. 4: Normalized GSR-ECG data of a person acquired during the training phase.

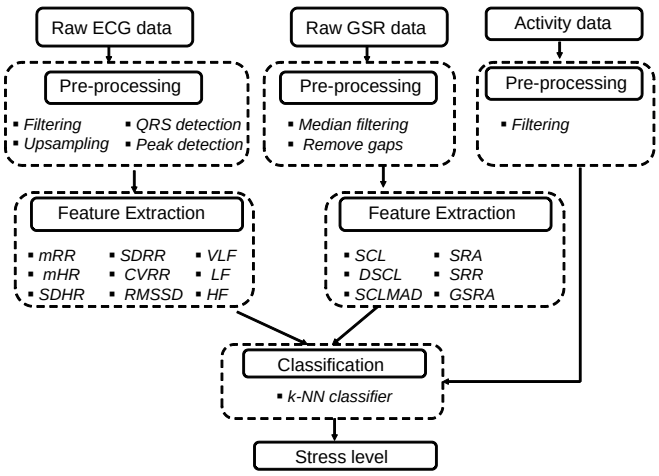


Fig. 5: Stress detection model (workflow) for period T_C.

Training Phase: In the training phase we record GSR and ECG data for each person in the team. We consider the stress experienced by a person into three categories, namely, low, medium, and high. We extract different features from GSR and ECG data for different levels for stress for each person in the team. To collect data for different levels of stress we have designed an experiment setup as shown in Table I. The experiment is divided into four phases where each phase consists of a set of questions and the difficulty level of questions increase as we move to the higher phases. A phase can be further divided into stages. Phase-2 and 3 are stress-inducing phases and consists of different stages, where each stage involves questions of a particular type like memory based, arithmetic, and logical questions. The first phase, Phase-1, starts with a relaxed stage (RX1) where the team members are asked to take deep breaths and relax, typically lasting for two minutes. The next phase Phase-2 (STR1) involves three stages (Stroop Test, Logical Questions, Arithmetic Questions). In Phase-2 (STR1) master gives Stroop test to team members, where each member says the name of the color of the word on the screen and not what the word says. This stage typically lasts 1 minute. Next, 3 logic and 3 arithmetic questions are given by the master to the team members. Each stage lasts for about nine minutes. In this phase the questions are simple and ample time is given to the team members. This completes Phase-2 of the experiment. In the next phase, Phase-3 (STR2), the difficulty level of questions increases and also the time allocated for each question is decreased. The first stage involves a memory-based question where team members are shown a sequence of colors for a minute and, at the end of the minute, they repeat the sequence back. The second stage is the Stroop test. In the next stage, a set of logic and arithmetic questions are asked. This stage typically lasts for seven minutes. In the next phase, Phase-4, the members are asked to relax (RX2). This phase typically lasts for three minutes. In our experiments we consider the acquisition time period (T_A) to be up to 1 minute and computation time period (T_C) to be up to 30 seconds. The team consists of 5 people and the device pool includes around 8-10 devices (static and mobile devices). The data rate is up to 4 Mbps for WiFi and up to 3 Mbps for Bluetooth 2.0, required for communication between sensors and smartphones.

Figure 4 shows normalized GSR data and normalized beats per minute (bpm) metric derived from ECG data of a person. Both GSR and ECG data are collected during different stages of the experiment (shown in Table I) conducted during the training phase to extract features of different levels of stress. We consider RX1 and RX2 to be low stress inducing phases, STR1 to be medium stress inducing phase, and ST2 to be high stress inducing phase. We observe significant increase in both GSR and bpm data in the two phases STR1 and STR2. Also, we observe that the value of both GSR and bpm increase as we move from STR1 to STR2 phase. In the relaxed stage RX1, on the other hand, we observe relatively low values of data and in RX2 we observe significant decrease in both GSR and bpm data after the two stress inducing stages.

Workflow: In Fig. 5 we present our workflow which shows the sequence of tasks that need to be executed to quantify the level of stress experienced by an individual. We first obtain the physiological data (ECG, GSR, and activity data) from the sensor nodes attached to a person's body. Next,

we preprocess the data to remove any noise in the signal. Preprocessing of ECG signal involves filtering the data to remove noise, upsampling, QRS complex detection, and peak to peak detection (involves estimation of R-R interval in the QRS complex). The GSR signal contains two-sided local noise peaks that are caused by a physical disturbance of the contact between the skin and the sensors, this is removed via median filtering. Also, gaps in the GSR signal can be observed when the fit between the skin and the sensors is not tight enough, as the result the contact is continuously broken. This is done by checking the data for contiguous blocks of gaps and removing them. Activity data (accelerometer or annotations from the user) are filtered to remove noise. Next, step involves feature extraction from the pre-processed data. For ECG data, we extract time-domain features such mean RR interval (mRR), mean heart rate (mHR), heart rate deviation (SDHR), RR interval deviation (SDRR), coefficient of variance of RR interval, root mean square successive difference (RMSSD), as well as frequency-domain features such as power spectrum of very low frequency (VLF), power spectrum of low frequency (LF), and power spectrum of high frequency (HF). Features extracted from GSR data include mean skin conductance level (SCL), skin conductance level deviation (DSCL), mean absolute deviation of the skin conductance level (SCLMAD), number of GSR responses (SRR), amplitude of GSR responses in a window (SRA), sum of the area of GSR responses in a window (GSRA). A supervised learning algorithm, such as the K-Nearest Neighbor method (k-NN) [3], is employed for real-time stress estimation of the people in the group. Once the stress of each member of the team has been estimated, then the direction of propagation of stress in the team is quantified via G-Causality.

B. Distributed Computing Framework

The execution of stress detection algorithms can be executed locally in the personal mobile device of the team members or at a centralized location after receiving physiological data from all the team members. However, both these strategies have their disadvantages. Firstly, mobile devices have very limited battery capacity and executing stress detection algorithms for extended period of time entails additional energy expenditure of the battery besides executing their essential functionalities (e.g., calling and texting). Secondly, centralized execution suffers from a single point of failure. Also, in case the network data rate is low or the network connectivity suffers from frequent disconnections then significant overhead is incurred by mobile devices (in terms of time and energy) in communicating application data to the centralized computing resource.

Computation and communication overhead: In Fig. 6(a), we show how the residual capacity of different mobile devices varies over time when the stress detection algorithm are executed on them. All devices start with 100% battery capacity. We see that the two smartphones (HTC Desire and Motorola Atrix) have discharge up to 60 % in less than 5 hours when only stress detection algorithms are executed on them; this shows that the battery capacity of a single mobile device is not enough to monitor stress of an individual. In Fig. 6(b) we observe the energy expenditure incurred at different devices of the people in the group when stress of a group of people is monitored at a centralized node. The energy expenditure incurred at the devices here is due to the communication overhead incurred when the data is transferred from them to a centralized location. The stress detection algorithm is executed at the centralized location. To overcome the obstacle of limited energy faced by such devices, we envision that *pool of mobile devices* (laptops, smartphones, notebooks, and tablets) in the vicinity can collaborate to form an *elastic* resource pool. Such distributed pool provides its computation and communication capabilities so to enable applications that otherwise could not be supported because of limited battery and memory constraints of a single device. To achieve our vision, we propose a distributed computing framework that utilizes the local resources for data generation and processing, and delivers application results in near real time without relying on any external remote resource.

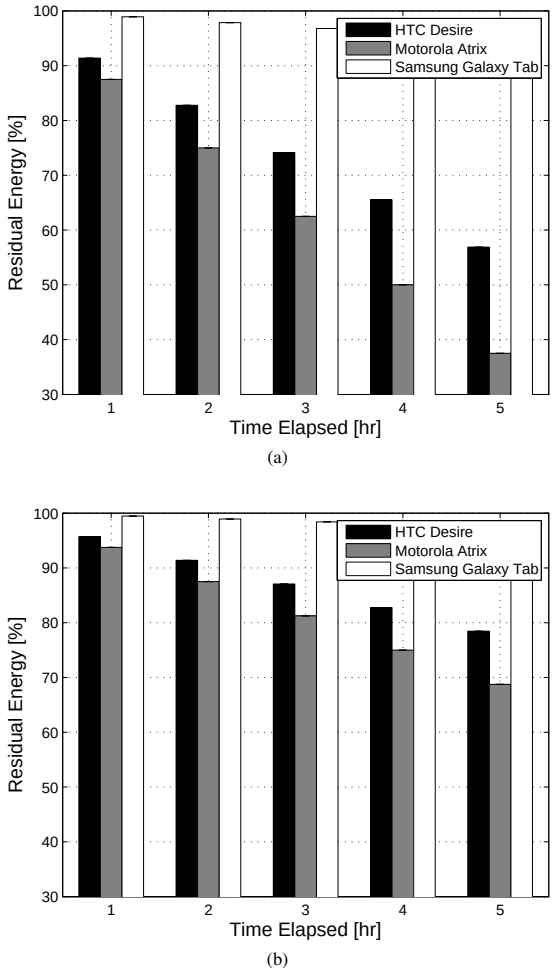


Fig. 6: (a) *Computation overhead* incurred at various mobile devices when stress is monitored for extended periods of time; (b) *Communication overhead* incurred when the physiological data is transferred from the local nodes to a centralized node.

Framework entities: The entities (i.e., the mobile devices in the vicinity) of the distributed computing framework may

at any time play one or more of the following three logical roles: i) *service requester*, which places requests for workloads that require additional data and/or computing resources from other devices, ii) *service provider*, which can be a data provider, resource provider, or both, and iii) an *arbitrator* (usually, the base station), which processes the requests from the requesters, determines the set of service providers that will provide or process data, and distributes the workload tasks among them. The service requester offloads (shares) the task of executing compute-intensive algorithms to (with) the service providers by submitting service requests to one of the arbitrators. Resource providers lend their computational (CPU cycles), storage (volatile and non-volatile memory), and communication (i.e., network interface capacity) resources for processing data. The arbitrator is aided by an energy-aware resource allocation engine that distributes the workload tasks optimally among the service providers.

Our framework applies to applications exhibiting *data parallelism* as well as to applications exhibiting *task parallelism*. Data-parallel applications are also referred to as "embarrassingly parallel" applications in which an independent set of tasks, homogeneous but working on disjoint sets of data, can be performed in parallel (but preceded and succeeded by pre- and post-processing tasks, respectively). Task-parallel applications, on the other hand, have a set of sequential as well as parallel tasks with pre-determined dependencies and degree of parallelism. Also, our framework is endowed with several *autonomic* capabilities such as self organization, self optimization, and self healing. The self-organization capability (for handling service discovery and service request arrivals as well as for task distribution and management) is imparted by the role-based architectural framework. It also facilitates interactions among the mobile entities for coordination and seamless switching among the three logical roles, namely, service requesters, service providers, and arbitrators [25].

Service discovery: Service discovery at the arbitrator is achieved through service advertisements from the service providers. Service advertisements will include information about the current position, amount of computing (γ_n^{cpu}, in terms of normalized CPU cycles), memory (γ_n^{mem} [Bytes]), and communication (γ_n^{net} [bps]) resources, the start (t_n^{in}) and end (t_n^{out}) times of the availability of those resources, and the available battery capacity (e_n^{adv} [Wh]) at each service provider n. The arbitrator is aware of the instantaneous power drawn by the workload tasks of a specific application when running on a specific class of CPU and memory (together given by c_n^{comp} [W]) as well as network (c_n^{net} [W]) resources at each service provider as the information about the different types of devices is known in advance. The arbitrators use the information from service advertisements of the N computing devices to derive the following: $\mathbf{R}$=$\{r_{mn}\}_{N\times N}$ [m], which conveys the distance r_{mn} between devices m and n, $\overline{S}$=$\{s_n\}_{1\times N}$, where $s_n \in \{1, 0\}$, which conveys whether n is a resource provider or not, and $\overline{D}$=$\{d_n\}_{1\times N}$, where $d_n \in \{1, 0\}$, which conveys whether n is a data provider or not.

Workload management: Each arbitrator is composed of two components, namely, *workload manager* and *scheduler/optimizer*. The workload manager handles tracking of workload requests, allocation of workload tasks, and aggregation of results. The optimizer identifies the number of service providers available and determines the optimal distribution of workload tasks among them. When a service requester needs additional computing resources to process the data it generates, it submits a service request to the nearest arbitrator and also specifies the maximum duration for which it is ready to wait for a service response. The optimizer will share the workload submitted by the requester among the available service providers based on one of several possible policies.

Energy-aware resource allocation: To take into account the different resource capabilities (in terms of residual battery capacity) and leverage heterogeneity in the vicinity, our framework incorporates an energy-aware resource-allocation engine. This engine selects the SPs and the number of tasks that should be given to each SP. An arbitrator is aware via advertisements of the residual battery capacity of all devices in the resource pool. In this formulation, the objective of the arbitrator is the *maximization of minimum residual battery capacity* at all the service providers, $\max\min_n\ e_n^{res}$ [Wh], while ensuring that the service response is delivered within δ^{max} [s].

$$\textbf{Maximize:}\ \min_n e_n^{res}, \tag{1}$$

$$\texttt{where},\ e_n^{res} = e_n^{adv} - (e_n^{data} + e_n^{comp}); \tag{2}$$

$$e_n^{data} = \frac{\delta_n^d}{3600} \cdot c_n^{net}; \tag{3}$$

$$e_n^{comp} = u_n \cdot \frac{\delta_n^s}{3600} \cdot c_n^{comp}. \tag{4}$$

In (2), $e_n^{data} + e_n^{comp}$ is the amount of battery capacity drained at each service provider n; δ_n^d for a service provider n depends on the amount of data it has to transmit (ω [Bytes] as a data provider) or aggregate ($\omega \cdot \sum_{i=1}^{N} a_{in}$ [Bytes] as a resource provider), and the availed communication capability, given by,

$$\delta_n^d = \begin{cases} f(\omega, \gamma_n^{net}) & \text{if } u_n = 0, \\ f(\omega \cdot \sum_{i=1}^{N} a_{in}) & \text{if } u_n = 1. \end{cases} \tag{5}$$

C. Propagation of Stress via Granger Causality

Granger-Causality is implemented as a part of the comprehensive stress detection algorithm (during T_R) to estimate the source of stress in a group of people as well as the direction of its propagation in a group. A time series $\mathbf{x} = \{x_1, x_2, \ldots, x_t, \ldots\}$ is said to Granger-cause another time series $\mathbf{y}$ if including information about the past of $\mathbf{x}$ significantly increases the prediction accuracy of the current value y_t of $\mathbf{y}$ in comparison to predicting it based *only* on the past values of y alone [6]. G-Causality was initially introduced in [10], where the authors implemented it using two vector Auto-Regressive (AR) models.

The first, called *restricted model*,

$$x_t = \sum_{j=1}^{P} a_j x_{t-j} + \delta_{1t}, \quad y_t = \sum_{j=1}^{P} a_j y_{t-j} + \gamma_{1t}, \tag{6}$$

calculates how much two time series, $\mathbf{x}$ and $\mathbf{y}$, can be 'explained' by their own past ($x_{t\text{-}j}$ and $y_{t\text{-}j}$, with $j = 1, 2, \ldots$), resulting in residual error variances $\Delta_1 = var(\delta_{1t})$ and $\Gamma_1 =$

$var(\gamma_{1t})$ (the model order is represented by P, which specifies how many previous time points are taken into account, and the length of the time series by T, with $P < T$).

In the second model, called *unrestricted model*,

$$\begin{aligned} x_t &= \sum_{j=1}^{P} a_j x_{t-j} + \sum_{j=1}^{P} b_j y_{t-j} + \delta_{2t}, \\ y_t &= \sum_{j=1}^{P} a_j y_{i-j} + \sum_{j=1}^{P} b_j x_{t-j} + \gamma_{2t}, \end{aligned} \tag{7}$$

the prediction is based on the time series' own past *and* the past of the other time series. This results in residual error variances $\Delta_2 = var(\delta_{2t})$ and $\Gamma_2 = var(\gamma_{2t})$. The linear influence from $\mathbf{x}$ to $\mathbf{y}$, annotated as $\mathcal{F}_{\mathbf{x}\to\mathbf{y}}$, and from $\mathbf{y}$ to $\mathbf{x}$, i.e., $\mathcal{F}_{\mathbf{y}\to\mathbf{x}}$, can now be calculated as the ratio between the variances of the residual error, i.e.,

$$\mathcal{F}_{\mathbf{x}\to\mathbf{y}} = \ln \frac{var(\gamma_{1t})}{var(\gamma_{2t})} = \ln \frac{\Gamma_1}{\Gamma_2}, \quad \mathcal{F}_{\mathbf{y}\to\mathbf{x}} = \ln \frac{var(\delta_{1t})}{var(\delta_{2t})} = \ln \frac{\Delta_1}{\Delta_2}. \tag{8}$$

A reduction in error variance when including the past of another time series results in a larger $\mathcal{F}$-ratio. The difference G-Causality, i.e., $\mathcal{F}_{\mathbf{x}\to\mathbf{y}}$–$\mathcal{F}_{\mathbf{y}\to\mathbf{x}}$, is then calculated to assess the *dominant* direction of information flow.

Selection of "time lag:" Selecting the time lag is an important problem to compute the G-Causality. The estimation of AR models requires as a parameter the number of time-lags P to include, i.e., the model order. Note that the decision on the model order is critical as too few lags can lead to a poor representation of the data, whereas too many of them can lead to problems in the model estimation [18]. Two criteria have been introduced in the literature, namely the Akaike Information Criterion (AIC) [2] and the Bayesian Information Criterion (BIC) [17], in order to estimate the model order P. Both these criteria help determine the quality of the model.

For n variables we have,

$$\begin{aligned} AIC(P) &= \ln |\mathbf{\Sigma_2}| + \frac{2Pn^2}{T}, \\ BIC(P) &= \ln |\mathbf{\Sigma_2}| + \frac{\ln(T)Pn^2}{T}. \end{aligned} \tag{9}$$

where $\mathbf{\Sigma_2}$ is the noise covariance matrix of the unrestricted model with $|\cdot|$ indicating the determinant of a matrix and T is the total number of datapoints used to fit the model to the data. The first term in (9), i.e., the logarithm of the determinant of the estimated noise covariance matrix, $\ln |\mathbf{\Sigma_2}|$), gives the prediction error for a model of order P, while the second terms in both the models serve as a 'penalty.' Both AIC and BIC differ as to how severely they penalize high-model orders, with BIC more heavily penalizing higher model orders than AIC. Either AIC or BIC are calculated for a set of model orders and the order that gives the minimum value of AIC or BIC is selected as the order of the AR model to determine G-Causality between two time series. The model with the lowest value of AIC indicates that it is the best model, i.e., it fits the data at hand better among all the models specified.

Distributed G-Causality: To enable distributed G-Causality calculation, we design a workflow that indicates the different tasks that have to be executed *in parallel* as well as the *sequential* steps that have to be taken to determine G-Causality. Figure 7 shows the workflow diagram to compute G-Causality. The G-Causality workflow takes as input the output of the stress workflow model, which estimates the stress experienced by each member of the team measured over acquisition period (T_A). In the first stage of the G-Causality workflow (Stage 1), as seen in Fig. 7, each sensor node performs pre-processing steps and checks the *stationarity* of the stress data [10]. In case the data is not stationary, the sensor nodes determine the time window over which the data can be assumed stationary, i.e., when the joint probability distribution does not change when shifted in time (and when, consequently, parameters such as mean and variance also do not change over time and do not follow any trends). In the second stage (Stage 2), each sensor node solves multiple linear regression problems, which aim at determining the model order as in (9). Next, each node receives data from all the other nodes in the group and calculates the model order from each pair of nodes (Stage 3). As the last stage (Stage 4), each node determines the G-Causality for each pair of group members.

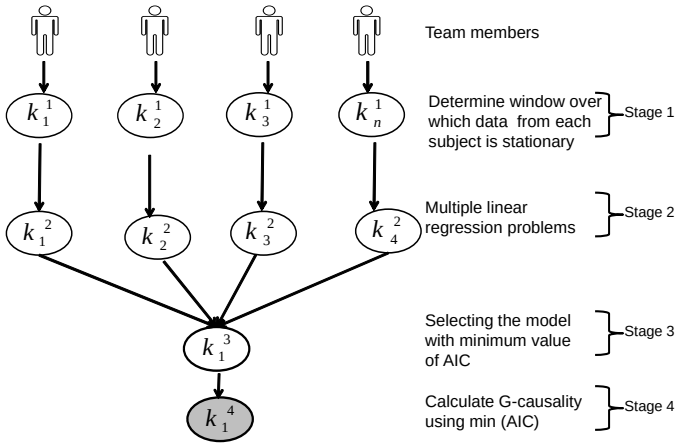
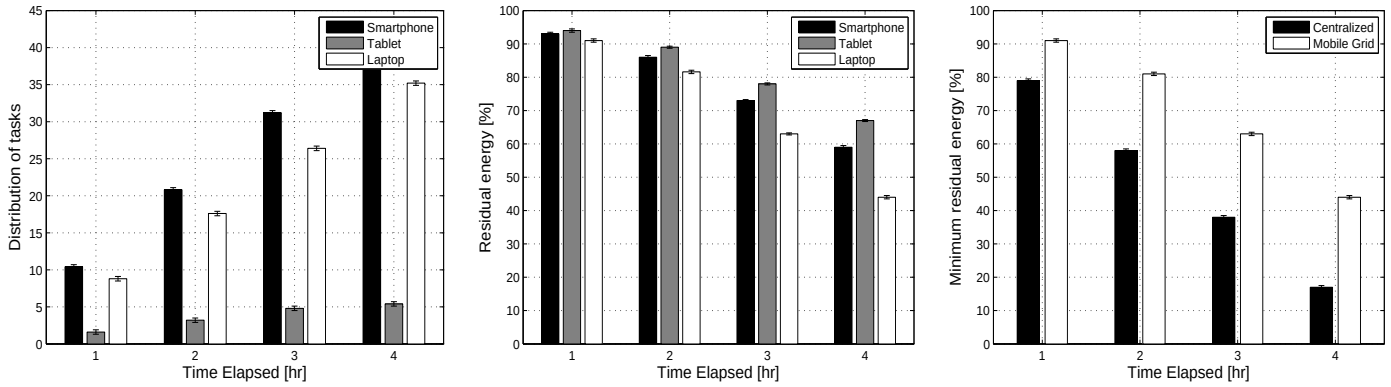


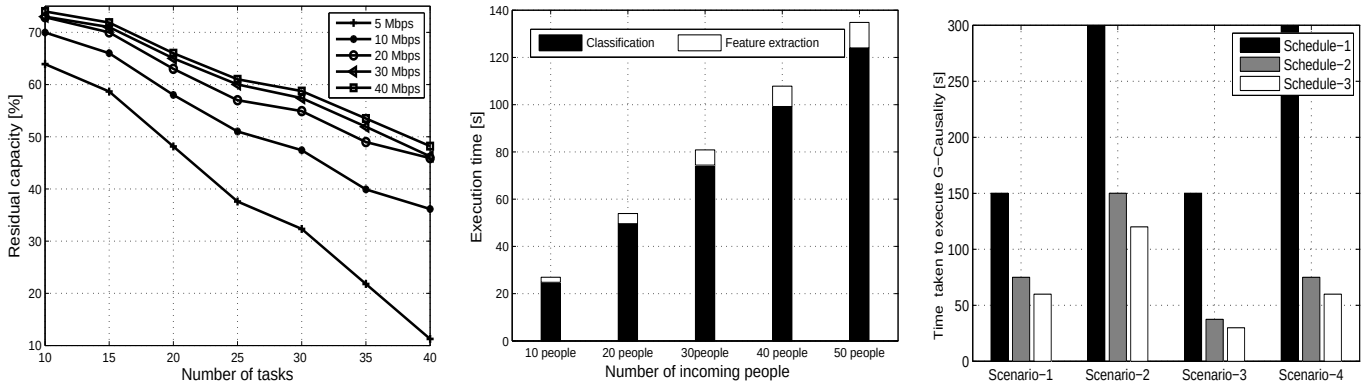
Fig. 7: Workflow designed to execute G-Causality in our distributed computing framework (*note that the output of the stress workflow serves as input to the G-Causality workflow*).

Note that the computation complexity to determine the stress for a team increases *linearly* with the number of team members. On the other hand, the computation complexity to determine the magnitude of G-Causality for a team increases *quadratically* with the number members. This is because we estimate G-Causality pair-wise, i.e, we calculate G-Causality between every two members of the team. To determine G-Causality between any pair of team members, the compute-intensive task is the estimation of time lag, as this requires solving a linear regression problem. The dominant operation in a linear regression problem is matrix inversion, whose complexity is given by $\mathcal{O}(L^3)$, where L is the length of the stress data of an individual. This shows that the computation time complexity of G-Causality between any two individuals increases by a cubic factor with the length of the stress data of an individual.



(a) (b) (c)

Fig. 8: (a) Distribution of tasks assigned to different SPs in our testbed over the course of time; (b) Residual energy [%] of the SPs in the resource pool over time; (c) Performance of our distributed mobile grid versus centralized execution in terms of minimum energy [%] in the testbed. The results with their 95% confidence intervals are plotted.



(a) (b) (c)

Fig. 9: (a) Variation of performance of distributed mobile grid in terms of minimum residual capacity [%] in the resource pool by varying different WiFi speeds; (b) Time time on a mobile device from our testbed (Nexus 5) for centralized online training of new people joining a team; (c) Time taken for the execution of G-Causality by varying time lag and number of service providers using different scheduling approaches.

IV. Performance Evaluation

We provide in this section the details about our testbed and our experiment methodology; then, we present our experiment scenarios, which are aimed at detecting stress of individuals in a team as well as the direction of propagation of stress among the members of team. We quantify the benefits of our proposed distributed computing framework to enable real-time stress detection for a group of people. We study how network connectivity impacts the performance of the distributed grid, and present on-line training of stress data as a candidate for distributed computing. Next, we explain our experimental setup to estimate direction of propagation of stress in a group. We provide observations and inferences from the experiments conducted, which corroborate the use of G-Causality as a tool to estimate direction of propagation. We also explain how our framework can implement distributed G-Causality and discuss the benefits obtained via our framework.

Experimental testbed: The testbed devices used in the experiment are: (i) HP Pavilion with intel i7 processor, 8 GB RAM and battery capacity of 4400 mA; (ii) Samsung Galaxy tablet running Android 2.2 with 1 MHz processor, 512 MB RAM and battery capacity of 4000 mAh; and (iii) Nexus 5 Smartphone running Android OS 4.4.4 with Quad Core 2.3 GHz–Snap dragon processor, 2 GB RAM and battery capacity of 2300 mAh.

TABLE II: **Confusion matrix for stress detection algorithm.**

		Predicted		
		Low Stress	Medium Stress	High Stress
Actual	Low Stress	3	2	0
	Medium Stress	1	4	0
	High Stress	0	0	5

Accuracy of stress detection algorithm: In Table II we show the confusion matrix for the stress detection algorithm. We consider three classes of stress low, medium, and high. We show the results for 15 different samples. We observe that the detection accuracy of high level of stress by the stress detection algorithm is the highest among all the classes. The F1 score which is given by $F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$, Precision $= \frac{tp}{tp+fp}$, and Recall $= \frac{tp}{tp+fn}$, where, tp is the number of true positive,

fp is the number of false positive, and fn is the number of false negatives. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. From the confusion matrix we derive the following scores for our experiments: for class low stress it is 0.67, class medium stress it is 0.73, and for class high stress it is 1.

A. Performance of Distributed Computing Framework

We present the performance of our distributed computing grid in terms of energy and number of tasks executed to estimate stress detection (executed in the time period T_C) of a group of people in comparison to a centralized approach. We also study the performance of mobile grid when the data rate of WiFi is varied.

Energy expenditure: In Fig. 8(a), we see the distribution of tasks among the different devices in the resource pool. The distribution of resources is done at the arbitrator. We see that different devices are given a different number of tasks over the course of monitoring stress. This depends on computational capability of devices and on their residual battery capacity. As a result, we see that different devices have been allocated different number of tasks over time. In Fig. 8(b), we see that the residual battery capacity of the devices in the resource pool is similar over a period of monitoring tasks. This is because the goal of the resource allocation in the proposed mobile grid is to maximize the minimum residual capacity of the resource pool and hence tasks are allocated to devices based on their residual battery capacity. In Fig. 8(c), we show the minimum percentage residual battery capacity of the resource pool in comparison to a centralized execution of the application, which is here shown for a Nexus 5 Smartphone in the testbed. We see that minimum percentage residual battery capacity of distributed mobile grid is always higher than that of centralized execution. After 2 and 3 hrs, we can see that the centralized execution has a lower energy than that of mobile grid by a 27 and 33%, respectively.

Impact of network connectivity: We vary the data rate of the local network to see its impact on the performance of distributed mobile grid. We consider four data rates 10, 20, 30, and 40 Mbps. We observe that, as the data rate reduces, the minimum residual energy [%] consumed increases. This is because, as the data rate decreases, it takes longer to transmit data to and receive results from the SPs, which consequently leads to an increased energy consumption of mobile devices. In Fig. 9(a) for 25 tasks and a data rate of 40 Mbps, we see that the minimum residual energy in the mobile grid is 16.39% lower than when the data rate is 10 Mbps. For 40 tasks and a data rate of 40 Mbps, we see that the minimum residual energy in the mobile grid is 25% lower than when data rate is 10 Mbps. From the figure we observe that for high data rates (around 30 Mbps and above) the performance of the mobile grid is very similar for different data rates.

On-line training: In Figure 9(b) we show that on-line training of new individuals in a team is also a good candidate for distributed mobile computing. For new individuals entering a team, the training and classification of stress data need to be done to monitor stress. We show in Fig. 9(b) that, as the number of people in the team increases, the execution time for classification also increases. Hence, in order to enable real-time stress detection of these new people in the team, the classification process needs to be done via distributed mobile computing.

B. G-Causality Results

Distributed computation of G-Causality: We compared the performance of centralized execution of G-Causality against its distributed computation using our mobile computing framework. The metric for comparison is the *time taken* to execute G-Causality for a team of people.

Experiments: We compared the centralized execution (where data is given from all nodes to a sink node for computation) (Schedule-1) with Round-Robin (in which we distribute an equal number of tasks to all the nodes) (Schedule-2), and a schedule where we distribute a different number of tasks to different nodes based on their computation capability (Schedule-3). We present how the time taken for executing group stress analysis varies as the number of resource providers, model order, and number of members in the group vary. We considered that the data is trasmitted from all sensors to the nearby devices via Bluetooth. The time taken by different devices to execute one unit task is a linear regression problem to estimate the model order. For more details on how devices are profiled and on the time taken to execute a unit task of G-Causality the interested reader is referred to [24].

Observations: Figure 9(c) shows the performance of the three scheduling approaches in terms of workload completion time. We see that the centralized execution (Schedule-1) as expected takes the maximum amount of time followed by Round-Robin (Schedule-2), whereas Schedule-3 takes the minimum amount of time. We divide our simulation into three scenarios: in Scenario-1, we assume the number of SP to be 5, the number of team members to be 5, and the maximum possible model order to be 10. In Scenario-2, we increase the maximum number of model order from 10 to 20, and in Scenario-3 we increase the number of service providers from 5 to 10 with everything else remaining the same. The result shows that the time taken is very sensitive to the model order. In Scenario-4 we vary both number of service providers and model order in order to compare the time taken by distributing tasks under different schedules. We see that Schedule-3 takes the least amount of time. For Scenario-1 and 3, the time taken by the centralized execution is not affected by the number of SPs and so it remains the same. Although both Round-Robin and Schedule-3 divide the task among SPs, Round-Robin performs worse than Schedule-3 because Schedule-3 assigns tasks to SPs based on their computational capability whereas Round-Robin distributes an equal number of tasks to all the nodes, irrespective of their capabilities.

Stress propagation: We now present results for stress propagation in a group. As discussed earlier, we use G-Causality to determine the extent and influence of stress from one group member to another. To quantify the mental group stress, we design our experiments to have four phases, as summarized in Table I.

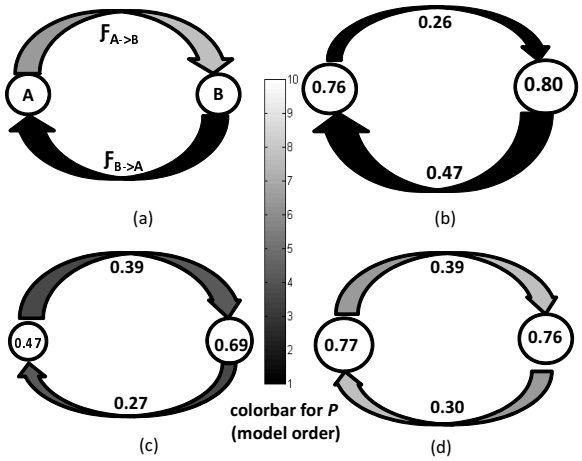


Fig. 10: G-Causality between two team members for different cases over multiple experiments. Note that the size of the node/circle indicates the stress experienced by each team member, the top clockwise edge label shows magnitude of G-Causality from team member A to B, and the bottom edge label shows the same from team member B to A; the edge color (gray scale) indicates the G-Causality time lag (model order) between stress data of team members (ranging from 1 to 10 in our experiments).

Experiments: In our analysis, a team consists of two members. One of the members of the team serves as a master and assigns tasks to the other member. This simplified configuration helps us understand the propagation of stress from master to the other team member. We perform group stress detection for three different cases: in Case-1, the first team member serves as a master and the other one (slave) and the master asks questions based on Table I. In Case-2 the second member serves as the master (i.e., they switch their roles in the experiment). In Case-3 both the two team members receive questions based on Table I from a laptop (or a person who is not part of the experiment). Case-3 helps us analyze the propagation of stress when each team member performs task independently. During an experiment, the master assigns questions to the slave(s), keeps track of time, and at the end of the experiment informs the slave(s) about his/her performance. We performed all the three cases thrice and in a random order to gain statistical insights. Each repetition of a case is termed as an 'experiment.'

We present our results of propagation of stress using an visually informative *bubble diagram*, as depicted in Fig. 10(a), where each member is represented by a bubble/circle. From any node i to j the following attributes exist: the *thickness* represents the magnitude of G-Causality from node i to j (clockwise edge); the *color* of the edge (gray scale) represents the model order number used to calculate the G-Causality; and the *size* of the bubble/circle represents the average stress experienced by the person over the course of experiment. The higher the influence of G-Causality, the thicker the edge; similarly, the higher the stress level experienced by a team member, the bigger the bubble size. We term the left node 'A' and the right node 'B' for the ease of understanding.

Then, we studied the influence of G-Causality between the two team members for different cases. We take the average of all the experiments conducted for each case to show the results for G-Causality. In Case-1, node B serves as master (examiner) and conducts the test for node A (examinee). We see that in this case, as shown in Fig. 10(b), the influence of G-Causality from node B to node A is higher than from A to B. Here, the dominant G-Causality direction is from B to A and the magnitude of influence is, on average, 0.47. As node B conducts the test, the dominant direction of G-Causality indicates that the examinee is influenced by the examiner, i.e., that the stress propagates from node B to node A. In Case-2, node A serves as the master (examiner) and node B serves as the slave (examinee), i.e., the roles are switched with respect to the previous case. We observe that in this case, as shown in Fig. 10(c), the influence of G-Causality from node A to node B is higher than the influence from node B to A. The magnitude of influence is on average 0.39. This indicates that node A being the examiner influences node B, i.e., that the stress propagates from A to B. In Fig. 10(d) both nodes A and B play the role of slaves and receive instruction from a computer (master), i.e., neither of the nodes interacts with each other. As a result, we see that the values of G-Causality from A to B and from B to A are very close. This indicates that neither of the nodes has a predominant influence on the other.

V. Conclusion and Future Work

We presented a real-time, in-situ stress detection for a group of people via a distributed computing framework. In the series of experiments we also studied the direction of propagation and magnitude of stress in a group, and analyzed how this stress propagates over time. The results of this analysis help quantify the direction in which the stress propagates in a group. The work presented enables taking real-time decisions and lays the foundation on how we can empower individuals who are in better condition (e.g., less stressed) by putting them dynamically, on a need-basis, in charge of a situation. This will help improve productivity in highly stressful situations like military operations by, for example, reorganizing dynamically hierarchy beyond existing ranks and roles.

As the next step, we plan to study other emotions besides stress, e.g., fear, anxiety, and anger and to develop a complete human mood elicitation system. This system will use other vital signs besides GSR and HR and will require processing a variety of models to estimate different emotions experienced by people in a team in real time. We also plan to extend our middleware to communicate with remote resources (such as Clouds) and access historical medical records of the people in the group in order to make more informed decisions. Last, but not least, we plan to conduct more experiments to understand how estimation of direction of propagation of stress in a team can be to used as a feedback to improve group dynamics in real time. We also plan to use additional methods e.g., mental imagery and virtual reality in the training phase to quantify stress of individuals. We also plan to integrate interrelationships between people in a group from social networking sites in our analysis of Granger Causality.

ACKNOWLEDGEMENTS

This work was supported by the DARPA Young Faculty Award (YFA) grant no. 11038087. The authors would like to thank Faiza Gazanfar, Rutgers ECE graduate student, for helping with the experiments.

REFERENCES

[1] Shimmer Sensors. http://www.shimmersensing.com/.
[2] H. Akaike. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.
[3] N. S. Altman. An Introduction to Kernel and Nearest-neighbor Nonparametric Regression. *The American Statistician*, 46(3):175–185, 1992.
[4] F. Angus, J. Zhai, and A. Barreto. Front-end Analog Pre-processing for Real-time Psychophysiological Stress Measurements. In *Proc. of the World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI)*, 2005.
[5] J. Bakker, M. Pechenizkiy, and N. Sidorova. What's your current stress level? Detection of Stress Patterns from GSR Sensor Data. In *Proc. of International Conference on Data Mining Workshops (ICDMW)*, Vancouver, Canada, December 2011.
[6] S. L. Bressler and A. K. Seth. Wiener–Granger causality: A Well Established Methodology. *Neuroimage*, 58(2):323–329, 2011.
[7] G. P. Cooke, J. A. Doust, and M. C. Steele. A Survey of Resilience, Burnout, and Tolerance of Uncertainty in Australian General Practice Registrars. *BMC Medical Education*, 13(1):2, 2013.
[8] A. De Santos, C. Sanchez-Avila, J. Guerra-Casanova, and G. Bailador-Del Pozo. Stress Detection by means of Stress Physiological Template. In *Proc. of World Congress on Nature and Biologically Inspired Computing (NaBIC)*, Salamanca, Spain, Oct 2011.
[9] A. Gaggioli, P. Cipresso, S. Serino, G. Pioggia, G. Tartarisco, G. Baldus, D. Corda, M. Ferro, N. Carbonaro, and A. Tognetti. A Decision Support System for Real-Time Stress Detection During Virtual Reality Exposure. *Studies in Health Technology and Informatics*, 196(A):114–120, 2014.
[10] J. Geweke. Measurement of Linear Dependence and Feedback between Multiple Time Series. *Journal of the American Statistical Association*, 77(378):304–313, 1982.
[11] J. Healey and R. Picard. SmartCar: detecting driver stress. In *Proc. on International Conference on Pattern Recognition*, Barcelona, Spain, Sept 2000.
[12] J. A. Healey and R. W. Picard. Detecting Stress during Real-world Driving Tasks using Physiological Sensors. *IEEE Transactions on Intelligent Transportation Systems*, 6(2):156–166, 2005.
[13] E. K. Lee, P. Pandey, and D. Pompili. Real-time Tracking of Stress Propagation using Distributed Granger Causality. In *Proc. of the ACM International Conference on Body Area Networks (BodyNets)*, Boston, MA, Sept 2013.
[14] M. Nkosi and F. Mekuria. Cloud Computing for Enhanced Mobile Health Applications. In *Proc. of International Conference on Cloud Computing Technology and Science (CloudCom)*, Indianapolis, IN, Nov 2010.
[15] M. Satyanarayanan. Fundamental Challenges in Mobile Computing. In *Proc. of ACM Symposium on Principles of Distributed Computing*, Philadelphia, PA, May 1996.
[16] M. B. Schippers, R. Renken, and C. Keysers. The Effect of Intra-and inter-subject Variability of Hemodynamic Responses on Group Level Granger Causality Analyses. *NeuroImage*, 57(1):22–36, 2011.
[17] G. Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461–464, 1978.
[18] A. K. Seth. A MATLAB Toolbox for Granger Causal Connectivity Analysis. *Journal of Neuroscience Methods*, 186(2):22–26, 2010.
[19] Q. Shen, X. Liang, X. Shen, X. Lin, and H. Luo. Exploiting Geo-Distributed Clouds for a E-Health Monitoring System With Minimum Service Delay and Privacy Preservation. *IEEE Journal of Biomedical and Health Informatics*, 18(2):430–439, March 2014.
[20] Y. Shi, M. H. Nguyen, P. Blitz, B. French, S. Fisk, F. De la Torre, and A. Smailagic. Personalized Stress Detection from Physiological Measurements. In *Proc. of International Symposium on Quality of Life Technology*, Iasi, Romania, June 2010.
[21] L. A. Simpson and L. Grant. Sources and Magnitude of Job Stress among Physicians. *Journal of Behavioral Medicine*, 14(1):27–42, 1991.
[22] F.-T. Sun, C. Kuo, H.-T. Cheng, S. Buthpitiya, P. Collins, and M. L. Griss. Activity-Aware Mental Stress Detection Using Physiological Sensors. In *Proc. of Intl. Conf. on Mobile Computing, Application, and Services (MobiCASE)*, Santa Clara, CA, October 2010.
[23] C. Tan, W. Chen, and G. Rauterberg. Interactive Aircraft Cabin Testbed for Stress-free Air Travel System Experiment: An Innovative Concurrent Design Approach. In *Proc. of International Conference on Advances in Mechanical Engineering (ICAME)*, Shah Alam, Malaysia, June 2009.
[24] H. Viswanathan, E. K. Lee, and D. Pompili. Enabling Real-time In-Situ Processing of Ubiquitous Mobile-Application Workflows. In *Proc. of IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Hangzhou, China, October 2013.
[25] H. Viswanathan, E. K. Lee, I. Rodero, and D. Pompili. Uncertainty-aware Autonomic Resource Provisioning for Mobile Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems*. preprints, 2014.
[26] H. Viswanathan, E. K. Lee, I. Rodero, and D. Pompili. An Autonomic Resource Provisioning Framework for Mobile Computing Grids. In *Proc. of IEEE International Conference on Autonomic Computing (ICAC)*, San Jose, CA, September 2012.
[27] Q. Xu, T. L. Nwe, and C. Guan. Cluster-Based Analysis for Personalized Stress Evaluation Using Physiological Signals. *IEEE Journal of Biomedical and Health Informatics*, 19(1):275–281, 2015.
[28] J. Zhai, A. B. Barreto, C. Chin, and C. Li. Realization of Stress Detection using Psychophysiological Signals for Improvement of Human-computer Interactions. In *Proc. of IEEE SoutheastCon Conference*, Ft. Lauderdale, FL, April 2005.

Parul Pandey is a PhD candidate in the Dept. of ECE at Rutgers U. She is currently working on mobile and approximate computing, cloud-assisted robotics, and underwater acoustic communications under the guidance of Dr. Pompili as a member of the Cyber-Physical Systems Laboratory (CPS-Lab). Previously, she had received her BS degree in electronics and communication engineering from Indira Gandhi Institute of Technology, Delhi, India and her MS degree in ECE from the U. of Utah, USA, in 2008 and 2011, respectively.

Eun Kyung Lee received his PhD in ECE from Rutgers University in 2015 after working in the CPS-Lab under the guidance of Dr. Dario Pompili. His research interests include energy-efficient data-center management and wireless sensor networks. Previously, he had received his BS in Electrical and Telecommunications Engineering from Soongsil University, South Korea and his MS in ECE from Rutgers University, in 2002 and 2009, respectively.

Dario Pompili is an Assoc. Prof. with the Dept. of ECE at Rutgers U. He is the director of the CPS-Lab, which focuses on mobile computing, wireless communications and networking, acoustic communications, sensor networks, and datacenter management. He received his PhD in ECE from the Georgia Institute of Technology in June 2007. He had previously received his 'Laurea' (integrated BS and MS) and Doctorate degrees in Telecommunications and System Engineering from the U. of Rome "La Sapienza," Italy, in 2001 and 2004, respectively. He is a recipient of the prestigious NSF CAREER'11, ONR Young Investigator Program'12, and DARPA Young Faculty'12 awards. He is a Senior Member of both the IEEE Communications Society and the ACM.