Chapter 7
Common Techniques for Self-Awareness and Self-Expression

Shuo Wang, Georg Nebehay, Lukas Esterle, Kristian Nymoen, and Leandro L. Minku

Abstract Chapter 5 has provided step-by-step guidelines on how to design self-aware and self-expressive systems, including several architectural patterns with different levels of self-awareness. Chapter 6 has explained important features in self-aware and self-expressive systems, including adaptivity, robustness, multi-objectivity and decentralisation. To allow such self-aware capabilities in each design pattern and enable those system features, this chapter introduces the common techniques that have been used and can be used in self-aware (SA) and self-expression (SE) systems, including online learning, nature-inspired learning and socially inspired learning in collective systems. Online learning allows learning in real time and thus has great flexibility and adaptivity. Nature-inspired learning provides tools to optimise SA/SE systems that can be used to reduce system complexity and costs. Socially inspired learning is inspired by common social behaviours to facilitate learning, particularly in multi-agent systems that are commonly seen in SA/SE systems. How these techniques contribute to SA/SE systems is explained through several case studies. Their potentials and limitations are widely discussed at different self-awareness levels.

Shuo Wang and Leandro Minku
School of Computer Science, University of Birmingham, UK.
e-mail: \{s.wang,l.l.minku\}@cs.bham.ac.uk

Georg Nebehay
Safety and Security Department, Austrian Institute of Technology, Austria.
e-mail: georg.nebehay.fl@ait.ac.at

Lukas Esterle
Institute of Networked and Embedded Systems, Alpen-Adria-University Klagenfurt, Austria.
e-mail: lukas.esterle@aau.at

Kristian Nymoen
Department of Musicology, University of Oslo, Norway.
e-mail: kristian.nymoen@imv.uio.no

7.1 Introduction

Chapter 5 has provided step-by-step guidelines on how to design self-aware and self-expressive systems, including several architectural patterns with different levels of self-awareness. Chapter 6 has explained important features in self-aware and self-expressive systems, including adaptivity, robustness, multi-objectivity and decentralisation. To allow such self-aware capabilities in each design pattern and enable those system features, this chapter introduces the learning methods that are widely applied and considered in SA/SE systems. First, online learning (Section 7.2) contains various techniques that can potentially encourage time awareness by updating learning models with time. Some online learning methods are designed to anticipate changes in learning data and environments, which may contribute to goal awareness and stimulus awareness. They also maximize system adaptivity, in order to overcome changing environments. Second, nature-inspired learning (Section 7.3) provides important ideas for designing large and complex systems. Meanwhile, it includes great techniques to realize multi-objectivity. Third, socially inspired learning (Section 7.4) includes a set of techniques for multi-agent systems, which can decentralise the system and reduce system complexity.

The above learning techniques are defined and explained in the following sections. In particular, example case studies are given to illustrate how they have been used in SA/SE systems at the levels of self-awareness. For each type of learning techniques, related learning methods that may potentially contribute to SA/SE systems are also introduced to widen the application for future use.

7.2 Online Learning

Online learning has been extensively studied and applied in the field of machine learning in recent years. Most machine learning methods operate in offline mode. They first learn how to perform a particular task and then are used to perform this task. No task can be performed during the learning phase and, after the learning phase is completed, the system cannot further improve or change [268]. However, a large number of real-world problems do not allow us to see all data in advance, and they are not intrinsically static. For example, consider an information filtering system which predicts the user's reading preferences. It is not possible to collect all users' information beforehand for learning. New users can join in and old users may leave at some point. Users can also change their preferences with time and a system which learnt how to predict them in the past will fail if it cannot update according to the new ones [358]. In a recommender or advertising system, customers' behaviour may change depending on the time of the year, on the inflation and on new products made available. Other examples include systems for computer security [140], market-basket analysis [12], spam detection [289] and web pages classification [8].

Differently from offline learning algorithms, online learning is brought up to perform a particular task at the same time as the learning occurs. Online learners

are updated whenever a new training example is available, being able to perform lifelong learning in the sense that they do not ever need to stop learning. So, they can attempt to adapt to possible changes in the environment, if properly designed [268]. In a more formal definition, online learning algorithms process each training example once "on arrival" without the need for storage and reprocessing, and maintain a model that reflects the current concept to make a prediction at each time step [307] [409]. The online learner is not given any process statistics for the observation sequence, and thus no statistical assumptions can be made in advance. A closely related concept to online learning is incremental learning. Incremental learning algorithms can also operate in changing environments, but training data are processed in chunks [268]. From another point of view, online learning can be viewed as a strict case of incremental learning, which processes data one by one. It can be used to solve both online and incremental problems. Due to its low time and memory requirements, besides being useful for applications in which training data arrive continuously (streams of data), online learning is also useful for applications with tight time and space restrictions, such as prediction of conditional branch outcomes in microprocessors [130].

Considering the nature and advantages of online learning, it contains useful techniques for SA/SE systems, as many SA/SE applications produce data continuously and the data may not be in a static state. For example, one of our applications is to perform object tracking and object detection in single-camera and multi-camera scenarios. In order to locate the object, the object detector must keep monitoring the location of targets in consecutive video frames. Meanwhile, the object tracking/detection can suffer from clutter and occlusion difficulties, as well as the execution time issue. Therefore, a fast online object detection method is adopted, to learn the appearance of objects in image streams and to re-detect the objects of interest after occlusions. More details on the application and the adopted techniques will be given in the following subsections.

7.2.1 Example Application

In object tracking, the general goal is to find the location of one or more objects of interest in consecutive frames of a video sequence, as it is depicted in Figure 7.1. Under some specific scenarios, training data may be available to aid the process of

Fig. 7.1 Tracking a single object of interest (in red).

recognizing objects. A learning algorithm can then be used in order to better un-

derstand the variability in the appearance of the objects of interest. In many cases, the object of interest is unknown beforehand and is indicated only by a single user-defined label that is typically given in the first frame of a sequence. Although no supervised training data is available, it is still often desirable to perform an update of the object model in order to reflect changes in the appearance of the object of interest. Many approaches model the problem of object tracking as binary classification in the online learning scenario. The two classes are the object of interest and background. We will now give a brief overview of existing techniques that fall into this category.

Collins et al. [76] were the first to employ binary classification in a tracking context. They employ feature selection in order to switch to the most discriminative colour space from a set of candidates and use mean-shift for finding the mode of a likelihood surface, thereby locating the object. In a similar spirit, Grabner et al. [146] perform online boosting and Babenko et al. [20] use multiple instance learning in order to find the location of the object. All of these methods use a form of reinforcement learning, meaning that the prediction of the classifier is directly used to update the classifier. While this approach enables the use of unlabelled data for training, it typically amplifies errors made in the prediction phase, thus leading to a degradation of tracking performance. In [147], this problem is addressed by casting object tracking as a semi-supervised learning problem, where only the first appearance of the object is used for updating. Both Kalal et al. [206] and Santner et al. [352] employ an optic-flow-based mechanism for labelling the available data in order to reduce the errors made in the prediction phase and demonstrate superior results. All of these methods employ online learning to incorporate new appearances of the object of interest into the object model with the aim of improving the general tracking performance.

In the following sections we will now explain the TLD approach of Kalal et al. in more detail. Section 7.2.1.1 gives a high-level overview of the approach. Section 7.2.1.2 introduces the learning component of TLD, a Random Fern classifier. Finally, Section 7.2.1.3 explains how online learning is performed in TLD.

7.2.1.1 Tracking-Learning-Detection

Kalal et al. [206] propose a solution to the tracking problem which they call *Tracking-Learning-Detection* (TLD). TLD consists of two separate components. The first component is a frame-to-frame tracker that predicts the location L_j of the object in frame I_j by calculating the optical flow between frames I_{j-1} and I_j and transforming L_{j-1} accordingly. Clearly, this approach is only feasible when the object is visible in the scene and fails otherwise. When the object is presumably tracked correctly (according to certain criteria) the location L_j is used in order to update the second component, which is a Random Fern classifier [308]. The Random Fern classifier is updated with positive training data from patches close to L_j and negative data from patches that exceed a certain distance. This classifier is then

applied in a sliding-window manner (see Figure 7.2) in order to re-initialize the frame-to-frame-tracker after failure.

Fig. 7.2 In TLD, a binary ensemble classifier is used to locate the object of interest by applying it in a sliding-window manner. The ability for multi-scale detection is achieved by scaling the size of the detection window. Image is from [282].

7.2.1.2 Random Fern Classifier

The Random Fern classifier [308] operates on binary features $f_1 \dots f_n$ calculated on the raw image data. These features are randomly partitioned into groups of so-called *ferns* $F_1 \dots F_m$ of size s

$$\overbrace{f_1 \dots f_s}^{F_1}, \overbrace{f_{s+1} \dots f_{2s}}^{F_2} \dots \overbrace{f_{(m-1)s+1} \dots f_{ms}}^{F_m}. \tag{7.1}$$

Ferns essentially are non-hierarchical trees, meaning that the outcome of each fern is independent of the order in which features are evaluated. The main reason for favouring ferns over trees is that they can be implemented extremely efficiently, an important property for real applications.

Suitable features for random ferns are proposed in [308], where a feature vector of size s consists of s binary tests performed on gray-scaled image patches. Each test compares the brightness values of two random pixels (See Figure 7.3). The

locations of the tests are generated once at startup and remain constant throughout the rest of the processing. The same set of tests is used with appropriate scaling for all subwindows. Input images are smoothed with a Gaussian kernel to reduce the effect of noise.

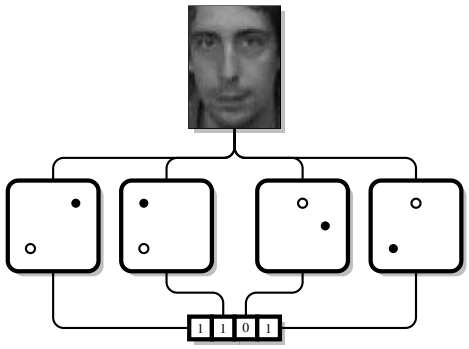

Fig. 7.3 Feature values depend on the brightness values of pairs of two random pixels. In this case, the outcome is the binary string 1101. Image is from [282].

7.2.1.3 Online Learning of the Random Fern Classifier

An important property of a classifier is the amount of time it takes to re-train the classifier as soon as new training data becomes available. In this respect, Random Ferns are superior to other classifiers such as Random Forests, as their formulation allows for a straightforward application in an online learning scenario. The posterior probability for each fern is

$$P(y=1|F_k) = \frac{P(y=1)P(F_k|y=1)}{\sum_{i=0}^{1} P(y=i)P(F_k|y=i)}, \tag{7.2}$$

where $y=1$ refers to the event that the object of interest is present in the subwindow. In TLD, the prior is assumed to be uniform, and the $P(F_k|y=i)$ are modelled as the absolute number of occurrences $\#p_{F_k}$ for positive training data and $\#n_{F_k}$ for negative training data. Therefore, the posterior probability becomes

$$P(y=1|F_k) = \frac{\#p_{F_k}}{\#p_{F_k} + \#n_{F_k}}. \tag{7.3}$$

When $\#p_{F_k} = \#n_{F_k} = 0$, then $P(y=1|F_k)$ is assumed to be 0 as well. The update procedure in TLD employs all training instances (i.e. all subwindows) for updating the classifier that are currently misclassified. A decision is obtained by employing a threshold θ on the posterior probabilities combined using the mean rule

$$\frac{1}{m}\sum_{i=1}^{m} P(y=1|F_i) \geq \theta. \tag{7.4}$$

7.2.2 Benefits and Challenges at Levels of Self-Awareness

In this section we examine TLD running on a single node with respect to the levels of self-awareness, pointing out benefits and drawbacks. It has to be noted that TLD runs on a single node.

7.2.2.1 Stimulus-Awareness

A stimulus causing the node to react refers to self-awareness on a very basic level. In TLD, this level of self-awareness is present in the computation of the optic flow that provides a predefined reaction to a certain optical stimulus provided by an image sensor. The reaction is in the form of computing the appropriate displacement vectors for the sparse motion field of the target. A very important aspect in this respect is the time that the node takes to react to the stimulus.
Benefits: In TLD, the employed method for optic-flow allows to compute the response tremendously fast.
Drawbacks: The optic flow component is unable to adapt itself.

7.2.2.2 Interaction-Awareness

While the virtual nodes of TLD in the form of ensemble members of the random fern classifier might be seen as virtual nodes, these nodes act completely independently, not performing any kind of interaction with the exception that their output is combined into a single response.
Benefits: By not modelling interactions between virtual nodes, computational efforts are kept low.
Drawbacks: Wrong decisions by individual virtual nodes are not identified.

7.2.2.3 Time-Awareness

Time-awareness is an enabling component for learning from experience, as errors made in the past are remembered and individual virtual nodes are adapted in order to prevent similar errors in the future. In TLD, time-awareness is present in the individual virtual nodes that each remember a history of misclassified feature values. It is however clear that this level of awareness comes at a certain computational cost.

Benefits: Time-awareness enables adaptation in order to avoid future errors.
Drawbacks: Time-awareness increases the complexity of a node considerably.

7.2.2.4 Goal-Awareness

The single goal in TLD is to track an object of interest as long as possible. While this goal is present implicitly in the adaptation of the posterior probabilities of the individual ferns, there is no explicit modelling of this goal.
Benefits: N/A
Drawbacks: N/A

7.2.2.5 Meta-Self-Awareness

There is an interesting feedback loop in TLD that we consider to render TLD meta-self-aware. This feedback loop refers to how the training examples in TLD are obtained, namely by extracting them by making use of its very own stimulus-aware component, the optic-flow-based tracker. The output of this optic flow-based tracker serves as a mean to distinguish between positive and negative training examples in the environment. and to adapt the behaviour of the node to improve in the future.
Benefits: Meta-self-awareness allows for the automatic extraction of training data.
Drawbacks: Absolute correctness of the extracted data is hard to verify.

7.2.3 Other Related Techniques

This section discusses other related techniques, namely ensemble learning, class imbalance learning, techniques for dealing with concept drift, and reinforcement learning. It does not provide a comprehensive overview of learning methods, but focuses on techniques that we believe to be more relevant to self-awareness and self-expression.

7.2.3.1 Ensemble Learning

Ensemble learning techniques have been given particular attention in online learning. The idea of ensemble learning is to employ multiple learners on a given problem and combine their outputs as a "committee" to make a final decision for better accuracy. The individual committee member is sometimes called base learner. In classification, ensemble learning is also referred to as multiple classifier system [166], classifier fusion [229], committee of classifiers, classifier combination, etc. The members' prediction might be real-valued numbers, class labels, posterior probabilities, or any other quality. To make the best use of the strengths of the indi-

viduals and make up their weaknesses, how to combine the predictions is important and has been studied extensively in the literature [361] [220].

Bagging [43] and AdaBoost [134] are the most popular offline ensemble algorithms in the literature, based on which numerous variations have appeared for different learning scenarios. There also exist other popular ones, such as Random Subspace [165], Random Forests [44] and Negative Correlation Learning [250].

Ensemble learning methods have some desirable features, which encourage the rapid growth of related research. For theoretical reasons [98], every single learning model has limitations and may perform differently due to insufficient data. Averaging many of them can reduce the overall risk of making a poor prediction [319]. It is an innate behaviour to consult opinions from others before a decision is made. Second, certain learning algorithms confront the local optima problem, such as neural networks and decision trees. Ensembles may avoid it by running local search from different views, where a better approximation to the true function is expected. Besides, some problems are just too difficult and complex which are beyond the learning ability of the chosen models. Ensembles allow partition of the data space, where each individual only learns from one of the smaller and simpler sub-problems. Their combination can then represent the whole problem better. For practical reasons [325], real-world problems can be very large or small. A large data set can be divided into several smaller subsets, which will be processed by multiple learners in parallel. In the case of too little data, resampling techniques can be used in ensembles for drawing overlapping subsets to emphasize the available data.

In addition to the above advantages, in the context of online learning, ensemble learning has become a preferable solution, because it has a more flexible and robust training framework than other single-model training methods, especially for non-stationary online scenarios. On one hand, building and maintaining multiple learners allows model updating without forgetting previously learnt information. Online Bagging [306] and Online Boosting [130] [306] are two successful online extensions of the well-established offline Bagging and Boosting. They have been widely used for processing static data streams. On the other hand, an ensemble model can be expanded for future data coming from a new data concept. For example, when processing image data streams, the image result can be affected by the machine capturing images and the environment where the machine is placed. If any change occurs, for a single-model learner, it has to be re-trained to learn the new data concept, whereas for an ensemble learner, a new learner can be simply trained based on the new data, which can then be added to the ensemble model considering that the existing learners in the ensemble may still contain some useful information. Section 7.2.3.3 further explains how ensembles can be used to deal with changes in data concept.

The aforementioned techniques can intrinsically facilitate learning tasks in online SA/SE systems. With regards to the five levels of self-awareness, it is apparent that the characteristic of real-time processing of online ensemble learning methods allows time-awareness. The time information in data streams can be utilized for time series modelling and/or anticipation. Furthermore, due to the flexibility and robustness to dynamic environments, online ensemble learning methods may encourage

goal-awareness. If the learning objective is changing overtime, appropriate ensemble techniques can be applied to sense the change and/or adjust the learning process correspondingly and adaptively.

7.2.3.2 Class Imbalance Learning

It is worth mentioning that there is a specific class of online ensemble learning methods, aiming to tackle imbalanced data streams. "Class imbalance" is a type of classification problems where some classes of data are heavily under-represented compared to other classes. It is commonly seen in real-world applications, such as intrusion detection in computer networks and fault diagnosis of control monitoring systems [407]. This type of data suffers learning issues caused by the relatively or absolutely under-represented class (minority) that cannot draw equal attention to the learning algorithm compared to the majority class. It often leads to very specific classification rules or missing rules for the minority class without much generalization ability for future prediction. This problem is aggravated when data arrive in an online fashion. Therefore, special treatments are required to overcome class imbalance.

When the received data become imbalanced, it is necessary for the relevant nodes in the SA/SE systems to have the knowledge of data status (seen as a type of stimuli), so that corresponding events can be triggered to maintain the system performance. Among very limited research that can enable this type of stimulus-awareness, Wang et al. developed a class imbalance detector that reports the real-time class imbalance status online [406]. Once class imbalance occurs, it will inform the running online ensemble learner, which can then adopt some state-of-the-art class imbalance techniques to adjust the learning bias. Based on the traditional Online Bagging, several resampling-based ensemble variations have been proposed, which apply oversampling or undersampling techniques to boost the role of the minority class and give online predictions, such as OOB, UOB [406], MOSOB [408] and WEOB [409]. Generally speaking, we expect the training framework of online class imbalance learning to introduce certain degree of stimulus-awareness into the ensemble learning framework with time-awareness and goal-awareness.

7.2.3.3 Concept Drift Techniques

As mentioned in the beginning of Section 7.2, many SA/SE applications produce data continuously and the data may not be in a static state, i.e., the joint probability distribution of the data may change with time. Such changes are referred to as *concept drifts* [139]. Online learning algorithms can implement strategies to deal with concept drifts, so that the detrimental effect of changes can be diminished and the learning models can recover from changes. As explained in Section 6.2.2.4, such algorithms can be based either on explicit change detection methods or on mechanisms that do not use explicit change detection. That section concentrated

on the advantages / disadvantages of using change detection methods. The current section concentrates on explaining existing online supervised learning algorithms themselves[1].

Algorithms with explicit change detection methods:

Change (a.k.a. drift) detection methods are usually based on the idea that the performance of a model improves over time when the data concept is stable. Therefore, such methods detect a concept drift when a considerable drop in the performance is observed. For example, the Drift Detection Method (DDM) algorithm [139] monitors the error rate and detects a concept drift if the error rate increases above the confidence interval of the minimum error rate so far. Early Drift Detection Method (EDDM) [22] monitors the average distance between any two misclassifications and detects a change if the current average distance is larger than the minimum distance so far by a pre-defined threshold. Statistical Test of Equal Proportions (STEPD) [290] monitors the difference in accuracy on older and more recent training examples and uses a statistical test to detect changes. These algorithms maintain a learning model which is reset upon change detection [139, 22, 290].

In order to reduce the problems caused by false positive change detections (Section 6.2.2.4), the algorithm Two Online Classifiers for Learning and Detecting Concept Drift (Todi) [289] maintains two classifiers in the learning system instead of one. Only one of these classifiers is reset upon change detection, so that the model representing the old data concept can still be used in the case of a false positive drift detection. Another approach able to avoid problems with false positives is Diversity for Dealing with Drifts (DDD) [269]. It maintains different ensembles to achieve better robustness to different types of changes. Among them, an ensemble representing the old data concept well is kept both for dealing with slow changes and false positive change detections. There are also algorithms to cope with recurring concepts, i.e., with concept drifts that take the current data distribution to a previously seen state. For instance, Just-in-Time Classifiers for Recurrent Concepts [9] keeps a representation for each concept that has been identified so far. Whenever the concept observed after a change detection matches an existing concept representation, a model created for that representation is retrieved so that this concept does not have to be learnt from scratch.

Algorithms with no explicit change detection method:

Most algorithms with no explicit change detection method are ensembles of learning machines specifically designed to deal with concept drift. These algorithms usually maintain a set of models which possibly represent different data concepts. New models are created to represent new concepts, whereas existing models deemed to be out-of-date can be deleted to gradually forget old concepts. Then, by emphasising the predictions given by the models more likely to perform well on the current data concept, these ensembles can deal with concept drifts.

[1] It is worth noting that there are also several other algorithms for dealing with concept drift in chunk-based learning (i.e., algorithms that are not online algorithms in the strict sense) [332, 72, 377, 118].

For example, Concept Drift Committee (CDC) [372] maintains an ensemble whose base models are weighted based on their accuracies. A new base model is created whenever a new training example is made available. When the maximum ensemble size is reached, a new model is added only if an existing model can be deleted. Models are deleted based on their current accuracy. Different from CDC, Dynamic Weight Majority (DWM) [224] creates new base models only at every p training examples, if the ensemble misclassifies the training example. This helps DWM to maintain a more diverse set of base models representing different concepts. Rather than assigning weights to base models according to their overall recent accuracy, Dynamic Classifier Selection [397] assigns weights to base models based on their local accuracy. Local accuracy is the accuracy on a subset of validation examples most similar to the example being predicted. Even though this algorithm may achieve better performance by tailoring weights to the specific example being predicted, a set of validation examples must the stored.

7.2.3.4 Reinforcement Learning

In self-aware systems reinforcement learning deals with nodes that are allowed to take certain actions in order to maximize an objective function. For instance, in the context of people tracking, one action for a camera node might consist of "hand over the current object to the next camera in the environment". The objective function in such a camera network might consider the balance between accurate tracking and the overall computational load of the camera nodes. A camera node might therefore decide to stop tracking the object in order to reduce the penalty imposed by its computational load when other cameras have a better view of the object. In order to be able to perform reinforcement learning, a node must be able to analyse its environment using its sensors as well as to interact with the environment using its self-expression capabilities.

The most striking difference between reinforcement learning and classical machine learning is the absence of labels. Instead of looking at examples (as in supervised machine learning), reinforcement learning algorithms perform a form of trial-error in order to explore its environment and find out which actions lead to a long-term increase of the objective functions. An intriguing problem is that the environment is constantly changing, possibly also due to the action of the node. For instance, handing over an object from one camera might be a good idea if the other camera is able to continue tracking the object. However, if the other camera quickly loses the object, the benefit of reducing the computational load is thwarted by the loss of tracking accuracy. Reinforcement learning algorithms usually employ some form of probabilistic model that are commonly based on Markov Decision Processes.

7.3 Nature-Inspired Learning

The field of nature-inspired learning is an inter-disciplinary area of research concerned with the problem-solving and computational capabilities of natural systems. Nature provides great sources for inspirations to both develop intelligent systems and gives solutions to complicated problems. For example, the analogy between the human nervous system and computational devices has been studied and exploited comprehensively from theories to application. It leads to the development of mathematical models of computation, such as neural networks [26]. The resulting techniques have contributed to substantial real-world applications successfully, especially in pattern recognition. Taking animals for example, evolutionary pressure forces them to develop highly optimised organs and skills to survive. Those organs and skills can be well refined as optimisation algorithms, and the evolution can be described as the process to fine-tune the parameter settings in the algorithms. Genetic algorithm (GA) and ant colony optimisation (ACO) are popular ones, belonging to this category. Simply to say, nature-inspired learning, namely natural computing, is the computational version of the process of extracting ideas from nature to develop 'artificial' (computational) systems, or using natural media (e.g. molecules) to perform computation [55].

Regarding the question of when to use nature-inspired learning approaches, some particular situations have been concluded [55]:

- The problem to be solved is complex, i.e., involves a large number of variables or potential solutions, is highly dynamic, non-linear, etc.
- It is not possible to guarantee that a potential solution found is optimal, but it is possible to find a quality measure that allows the comparison of solutions among themselves.
- The problem to be solved cannot be (suitably) modelled, such as pattern recognition and classification tasks.
- A single solution is not good enough.
- Biological, physical and chemical systems and processes have to be simulated or emulated with realism.
- Life behaviours and phenomena have to be synthesized in artificial media.
- The limits of current technology are reached or new computing materials have to be sought.

With the behaviour of natural systems, nature-inspired learning approaches are also shown and expected to possess greater adaptivity and resilience to applied environments than other learning systems. According to Marrow's work in 2000 [256], this adaptivity and resilience arise from several aspects:

- Because of the large number of elements in each system (which may be individual animals, cells or even macromolecules within a cell), each of them may be interchangeable for another.

- Because of the loose but flexible interconnections between elements, nature-inspired learning allows transfer of tasks between entities, and communication throughout the system.
- Because of the differences between elements in the system, nature-inspired learning allows a diversity of responses to a changing environment.
- The resulting complex environment that all the interacting parts produce stimulates diverse responses from living organisms.

The above understandings of nature-inspired learning approaches provide good reasons to solve complex problems in SA/SE systems that are usually formed of multiple SA/SE modules and software agents.

7.3.1 Example Application

We consider an example of a nature-inspired self-aware and self-expressive computing system within the field of *active music*. In an active music system computational nodes play malleable music that may be changed by a user or by the computational node itself (see further description and more examples of active music systems in Chapter 14). When a group of nodes play music together, most important challenges faced is timing. By sharing some common notion of time, the nodes may create various controlled musical patterns, such as rhythms or melodies.

In a setup where a collaborative active music system is implemented on mobile phones, the distribution of the system on separate units of the system calls for a mechanism for *decentralised* synchronisation. That is, there should not be a central unit to which the remaining nodes synchronise. A decentralised approach would allow nodes to leave or enter a musical performance at any time.

The example algorithm given in this section is inspired by fireflies who synchronise their flashes in a decentralised manner. The phenomenon has been studied for almost a century, and has more recently been proven useful when synchronising devices in machine-to-machine systems [37]. The algorithm allows pulse-based communication, and thus individual musical tones may be used for communication between nodes. The algorithm presented was introduced in [297], and is based on Mirollo and Strogatz' firefly algorithm [272] with various changes to compensate for delays in the system and faulty nodes, as will be explained in the coming section. More examples of how nature-inspired learning may be used to obtain self-awareness in active music systems is given in Chapter 14.

7.3.1.1 Fireflies as Inspiration for Pulse-coupled Oscillators

The phenomenon of certain species of firefly on riverbanks in Asia that synchronise their flashes has been observed and studied for more than a decade [149]. Throughout the 20th century John Buck proposed several explanations to the phenomenon

[47, 48], although a satisfactory mathematical model was not proposed until the last part of the century. Inspired by C.S. Peskin's work on heart physiology [320], Renato E. Mirollo and Steven H. Strogatz succeed in proving that the phenomenon could be described as *pulse-coupled oscillators* [272]. Their proof, however, required the fire events to be infinitely short impulses, with no delay in the communication, and also required that oscillator frequencies were equal.

In our example system, the synchronising units are musical nodes, each with an internal oscillator with a certain frequency (ω) and phase (ϕ). To enable interaction between such nodes, they must be *coupled*, either continuously by having each node monitor the phase of the oscillators of the other nodes (phase-coupling), or discretely though pulse-like communications (fire events) whenever an oscillator node reaches a phase threshold in its cycle (pulse-coupling). While phase-coupled systems provide continuous updates, and theoretical models show how they can be programmed to obtain synchronisation between oscillators, these systems are difficult to implement in real-world scenarios [37]. Pulse-coupled systems, on the other hand, are less difficult to implement in terms of inter-node communication, and as such Mirollo and Strogatz' firefly model is well suited for enabling synchronisation of active music nodes.

The fundamental operation of the algorithm described by Mirollo and Strogatz, is that the phase of an oscillator is increased by a small amount whenever a fire event is received from another oscillator. The amount by which the phase is increased depends on the internal phase of the receiving oscillator. It is imperative to their model that the larger the phase of the receiving node, the larger phase jumps should occur. The process is illustrated for two nodes in Figure 7.4.

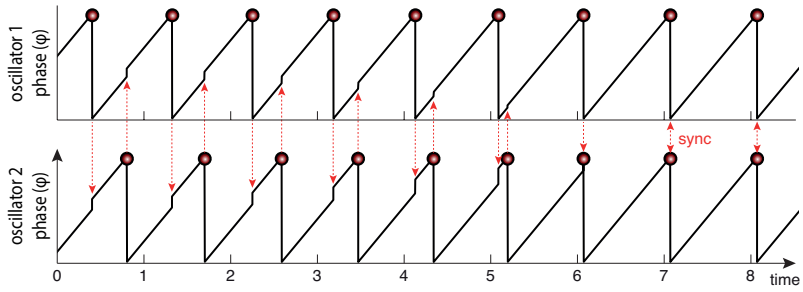

Fig. 7.4 Synchronisation of two firefly nodes by the Mirollo/Strogatz algorithm. The two nodes have identical frequencies, and there is no communication delay. Fire events are indicated by black circles, and the corresponding interaction indicated by a red arrow. Note how the size of the phase jump depends on the current phase of the receiving node.

Since Mirollo and Strogatz' proof, several efforts have been made towards reaching a synchronised state for real systems. Specifically, the requirements of infinitely short impulses and no communication delay is problematic in real systems. An increased effort has been seen in recent years, as decentralised computing systems have become more popular.

Mathar and Mattfeldt showed that the problem of transmission delays and transmission pulses of finite length can be tackled by implementing a *refractory period* directly after firing [260]. Similarly to how neurons enter a refractory state directly after firing, in which it is unable to receive new input, a pulse-coupled oscillator does not respond to fire events received within the refractory period.

Babaoglu et al. addressed the challenging scenario of synchronising clock cycles in certain types of sensor networks which lack a central reference clock to synchronize with [19]. A number of efforts have been made to tackle the challenge of "deafness" that occurs when a node is sending and receiving a fire event at the same time. Werner-Allen et al. [412], and Leidenfrost and Elmenreich [234] suggested the *Reachback firefly algorithm* for this purpose, where instead of making immediate phase jumps, the received impulses are accumulated during each period, and the corresponding phase jump is made at the beginning of the next cycle.

Klinglmayr et al. target the problem of robustness against faulty nodes in a system. That is, nodes that become defective, or malicious intruding nodes that may potentially disturb the operation of the network [221, 222]. Their approach is rather than pushing the phase of receiving nodes forward (excitatory coupling), to decrease the phase of the receiving node (inhibitory coupling).

7.3.1.2 Decentralised Phase Synchronisation in Active Music Systems

We use as example a collaborative active music system where each node is implemented on a mobile phone. The nodes communicate using audible sound. In other words, when reaching maximum phase a node outputs a short sound through its loudspeaker that other nodes pick up through their microphone. Node frequencies correspond to the duration of musical notes (eight notes, quarter notes, full notes, etc.). To exemplify, in a tempo of 120 beats per minute, a quarter note has a duration of 500 ms, a 16th note 125 ms, and a full note 2000 ms, corresponding to 2 Hz, 8 Hz and 0.5 Hz, respectively.

Given the low frequencies of the nodes in the application, the problem of transmission delay and finite duration of the sounds may easily be solved by implementing pulse-coupled oscillators with a refractory period. As descried by Klinglmayr et al. [221], the refractory period t_{ref} has a corresponding *refractory phase* interval $[0, \phi_{ref}]$ within which no phase jumps occur. Thus, when a node i not in a refractory state perceives a fire event from a node j, it immediately increases its own phase according to the *phase update function*, $P(\phi_i(t))$. More precisely:

$$\phi_j(t) = 1 \Rightarrow \begin{cases} \phi_j(t^+) = 0 & \\ \phi_i(t^+) = \phi_i(t^+) & \text{if } \phi_j(t) \in [0, \phi_{ref}] \quad \forall i \neq j \\ \phi_i(t^+) = P(\phi_i(t)) & \text{if } \phi_j(t) \notin [0, \phi_{ref}] \quad \forall i \neq j \end{cases}, \tag{7.5}$$

where t^+ is the time step immediately after t. The phase update function is given by:

$$P(\phi) = (1+\alpha)\phi, \tag{7.6}$$

where the *pulse coupling constant*, α, denotes the coupling strength between nodes.

One example of a musical system with the above implementation has been implemented on mobile phones using Mobmuplat [191] and Pd [328] at the University of Oslo.[2] In the example, each node has a fixed frequency of one hertz, phase coupling constant $\alpha = 0.1$, and the refractory period is set to 40 ms. The source code is available online.[3]

7.3.1.3 Obtaining Frequency Synchronisation Through Self-awareness

The section above discussed a well-known solution to the problem of synchronising decentralised musical nodes with *equal* frequencies. Synchronisation of nodes with different starting frequencies however, is a much more challenging task. In this section we will show how nodes are able to synchronise when higher levels of self-awareness are implemented.

To reflect that periodic patterns in a musical system may occur at multiple simultaneous levels (quarternotes, 16ths etc.) we define a new target state for the system: *harmonic synchrony*. Harmonic is borrowed from the concept of *harmonics* in various waveforms, where the frequency of every harmonic is an integer multiple of the lowest (fundamental) frequency. Correspondingly, harmonic synchrony is a state where the frequency of each node is element of $\omega_{low} \cdot 2^{\mathbb{N}_0^+}$, where ω_{low} is the lowest frequency of all nodes in the group. As an example, please refer to the illustration in Figure 7.5, where the most suitable adaptation would be if Node 1 adjusts to twice the frequency of Node 2, and Node 3 to four times the frequency of Node 2.

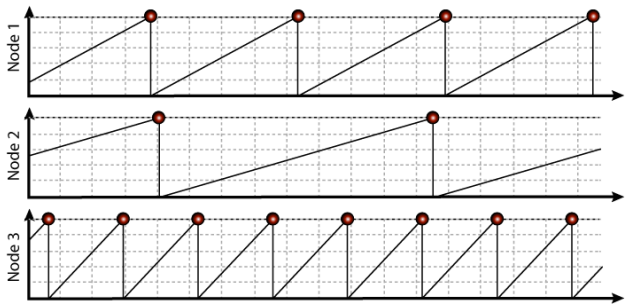

Fig. 7.5 Illustration of how *harmonic synchrony* may be beneficial in a collaborative active music system. The nodes may converge to integer-ratio frequencies, rather than equal frequencies.

In order to obtain synchrony, each node must be able to reason about its own level of synchrony with the rest of the group. A node calculates an error measure, $\varepsilon \in [0,1]$, whenever a fire event is received. ε is at its highest value when $\phi = 0.5$, and lowest value when ϕ is equal to 0 or 1.

[2] `http://vimeo.com/67205605`

[3] `http://fourms.uio.no/downloads/software/musicalfireflies`

$$\varepsilon = sin(\pi\phi(t))^2 \tag{7.7}$$

with the special case that $\varepsilon = 0$ if a fire event is received within the refractory period. The sequence of error measures calculated by a node make up a discrete function $\varepsilon(n)$ describing the error measures at the n-th received fire event. *Goal-awareness* is realised by applying a running median filter to $\varepsilon(n)$:

$$s = \text{median}\{\varepsilon(n), \varepsilon(n-1), ..., \varepsilon(n-m)\}, \tag{7.8}$$

where s is the self-assessed degree of synchrony within the node, and $m-1$ is the length of the median filter. Thus, s takes a high value when the node is out of phase with the past received fire events, and a low value when the node is in phase with the past perceived fire events. A node uses this self-assessment of synchrony to scale the impact of received fire events from other nodes upon its own frequency adjustments, and as such the goal awareness is the base of the node's *meta-self-awareness*. This self-awareness mechanism aids in giving less influence to occasional erroneous firings from other nodes or environmental noise.

Frequency adjustment is done as follows: Upon receiving a fire event from an external node, the receiving node calculates ρ, indicating whether frequency should be increased or decreased.

$$\rho = \sin(2\pi\phi(t)) \tag{7.9}$$

ρ is positive when $\phi < 0.5$ and negative when $\phi > 0.5$, meaning that if a node is less than half-way through its cycle when a fire event is received, it increases its frequency to "catch up" with the firing node.

Both ρ and s are factors in the frequency update function, where ρ indicates the direction and amount of change in frequency and s indicates the degree to which a node listens to or ignores incoming fire events. As such, s embodies the meta-self-awareness of firefly nodes.

To illustrate the operation of the algorithm, we look closer at a short excerpt from a single run. The example uses a Matlab simulation, showing a short excerpt from a run of 6 fireflies. Figure 7.6 shows how the phase coupling immediately pulls the phase of the oscillators closer together, for instance in the fire event of node 6 (yellow) after approximately 6.4 seconds. The phases of nodes 1, 2 and 4 are increased to maximum, causing them to reset, while node 5 is pulled back towards (but not fully down to) 0. Node 3 is approximately half-way through its cycle, and only a very small phase adjustment is made. The frequency coupling effect is shown between 7 and 8 seconds, where upon the firing of node 4 at 7 seconds, the phase of node 6 is being reset back to 0. In node 6, this is interpreted as being too early, and a negative frequency adjustment does take place the next time node 6 reaches its peak value (after $\sim$7.6 seconds). After approximately 12 seconds, the system reaches a close-to-synchronised state. That is, the oscillator frequencies are not quite in harmonic relation to each other, but due to the phase coupling, the fire events

occur as they should. A video showing the evolution of phases and frequencies of the fireflies is available online.[4]

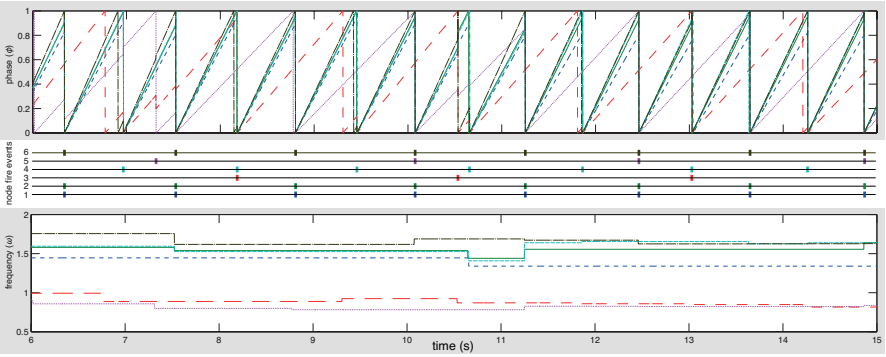

Fig. 7.6 A short excerpt from a Matlab simulation of 6 fireflies. The top plot shows the phases (ϕ) of all the nodes and the bottom plot shows the internal oscillator frequencies (ω). The fire events are shown in the middle plot which also acts as colour legend for the phase and frequency plots.

The chapter on active music systems presents several examples of how nature-inspired learning may be used to obtain self-awareness in active music systems. Please refer to Chapter 14 for examples of how the principle of pheromone trails of ants may be exploited to develop efficient gesture recognition algorithms and paths between musical sections.

7.3.2 Benefits and Challenges at Levels of Self-Awareness

The firefly algorithm could operate with different levels of computational self-awareness. In this section we analyse the behaviour of the algorithm given a certain level of self-awareness.

7.3.2.1 Stimulus-Awareness

A stimulus-aware firefly node is only aware of simple stimuli. The sender of fire events or the size of the group is unknown to each node. Each node in the example given above will react immediately to incoming stimuli (phase adjustment), and phase synchronisation may be realised at this level for nodes with equal frequencies.
Benefits: Simple and efficient implementation. No handshaking between nodes in a potentially large group.

[4] `http://vimeo.com/72493268`

Drawbacks: Nodes are unaware of their goal state and thus no self-assessment of synchrony is possible.

7.3.2.2 Interaction-Awareness

An interaction-aware firefly node is aware of the neighbouring nodes from which it receives fire events. For this to be possible, nodes must be identified using for instance a networking protocol.
Benefits: The sender of incoming fire events could be identified.
Drawbacks: Networking protocol required, which makes entering and leaving a musical interaction more difficult, especially if acoustic musical instruments are involved in the group.

7.3.2.3 Time-Awareness

Time-aware firefly nodes possess knowledge of previous events. Nodes are allowed to monitor a sequence of incoming fire events rather than only reacting immediately to a received fire event.
Benefits: Time-awareness enables the reachback firefly algorithm.
Drawbacks: Goal-awareness required for time filtering of error measure.

7.3.2.4 Goal-Awareness

A goal-aware firefly node is able to assess its own level of synchrony. This enables calculation of error measures for each received fire event, and in combination with time awareness becomes important in order to obtain meta-self-awareness.
Benefits: Reasoning about current degree of synchrony.
Drawbacks: Meta-self-awareness needed in order to change behaviour.

7.3.2.5 Meta-Self-Awareness

Meta-self-aware firefly nodes change their behaviour based on the knowledge from the other levels of self-awareness. Firefly nodes adjust their own sensitivity to fire events from other nodes based on their self-assessed degree of synchrony. Furthermore, nodes may use other frequency update mechanisms if they get "stuck" at a frequency where they never reach maximum phase [297].
Benefits: "Insecure" nodes make larger changes to oscillator frequency, keeping nodes in sync.
Drawbacks: Nodes are still only self-aware and not aware of internal parameters of other nodes.

7.3.3 Other Related Techniques

This section focuses on related techniques that we believe to be more relevant to self-awareness and self-expression, without meaning to be comprehensive.

Nature-inspired systems take inspiration from a natural process in order to obtain some goal. In our example above, the goal was to reach a certain *system-wide* state, which is often the case in such systems. Nature-inspired systems may take inspiration from how collective behaviour in certain species of animal reach a global goal. In the example above, the global goal of fireflies would be flashing in synchrony in order to enhance the light beyond that which each individual firefly is able to produce on its own. Variants of the firefly algorithm have been applied in a number of reports in order to synchronise devices in various types of self-aware and self-expressive system [412, 19, 70, 234]. A related algorithm has been used for a similar purpose, inspired by Japanese tree frogs [163]. The mating call of male Japanese tree frogs are not simultaneous, but rather spread out over time such that female frogs can locate the individual males.

Ant Colony Optimisation (ACO) is another type of nature-inspired learning useful for optimisation [106]. ACO takes inspiration from the way in which ants forage for food. In their search for food, ants lay behind a trail of pheromone for other ants to follow. When more ants pick the same trail, more pheromone is deposited, attracting even more ants. As such, trails to food sources become "ant highways" between the food source and the ant hill. When the food source runs out, the pheromone eventually evaporates, and new food sources are located. Dorigo et al. [107] showed how the use of a group of agents (ants) often would result in optimal solutions to problems without getting stuck at local optima. Examples of exploiting the pheromone mechanism from ACO in self-aware and self-expressive systems are given in Chapters 13 (multi-camera networks) and 14 (gait recognition in active music systems).

In optimisation, the field of Evolutionary Computing should also be mentioned in the context of nature-inspired methods. While several types of Evolutionary Algorithms (EA) exist, the common denominator is to search for a solution to a problem using mechanisms inspired by natural evolution [112]. An optimisation process is typically started by initialising a *population* of solutions to a problem. The individuals in the population are selected for reproduction through *recombination* and *mutation*, resulting in a offspring of which some survive and other perish based on some criterion. The process is repeated until some termination condition is met (e.g. if a satisfactory candidate has been evolved). Certain Evolutionary Computing techniques can also be used to deal with dynamic optimisation, i.e., when the environment, including the objective function, the decision variables, the problem instance, constraints and so on, may vary over time [427]. Techniques from Evolutionary Computing have been used for dynamic and multi-objective optimisation in SA/SE systems. Specifically, Chen et al. [61] applied a dynamic multi-objective evolutionary algorithm in a self-adaptive system for temperature management in multi-core FPGAs.

7.4 Socially Inspired Learning in Collective Systems

In multi-agent systems, the research on simulating societies in interaction with their environment has been taking shape, particularly the modelling of collective decisions. Collective (or distributed) learning is referred to as learning that is carried out by a group of agents instead of a single agent on its own, e.g. by exchanging knowledge or by observing other agents. It is an important concept in decentralising the system and reducing system complexity. It can be viewed as a social behaviour of the individual nodes in the collective. Having social behaviour allows to create a social network. This furthermore enables each individual to focus its efforts in making a collective decision towards this smaller set of entities in the collective. This reduces the complexity in a collective decision making process when resources are limited. This section will give a thorough explanation of socially inspired learning techniques in collective systems, through our SA/SE system example – the distributed smart camera network.

7.4.1 Example Application

In recent years 'dumb' cameras have evolved into embedded smart cameras [417, 343], combining a processing unit with an image sensor on a single platform. These processing capabilities, even though limited, allow the smart cameras to pre-process video data on-site and transmit only aggregated information, or a complete analysis of a scene, instead of plain images. Modern smart cameras are even capable of accomplishing processing intensive tasks, such as object tracking. In object tracking, a description of the object of interest is initially provided to the camera. The camera thereafter attempts to re-identify this object in consecutive frames of its own field of view (FOV). There are various tracking algorithms to locate objects in each frame matching the given description with the highest probability. While we employ tracking algorithms to identify and locate moving objects, we do not elaborate on these fundamental tracking techniques in this thesis.

Soon enough, single smart cameras have been connected to distributed smart camera systems [344, 338]. Tracking objects in multi-camera systems can be approached in two different ways. The first approach uses all cameras to track various objects and the gathered information is fused at a central control. When tracking objects within multiple cameras concurrently, cameras need to align their FOVs to ensure the gathered data is coherent. To do so, a calibration process is employed to remove geometric distortions caused by the camera lens. This calibration process requires additional work before the system can go online. This extra effort might be feasible with small numbers of cameras but could be highly problematic in larger systems with tens, hundreds or even thousands of cameras. Furthermore, in case one of the cameras' parameters is changed, new cameras are added or cameras are removed from the network, cameras might need to be re-calibrated to ensure proper functionality. While this might result in a very high network-wide utility, visual

object tracking is a resource intensive task and our facilitated cameras are rather resource constraint. We refer to the second approach as *distributed tracking*, where each object is only tracked by a single camera at any given time. This means, the collective of cameras has to know where each object currently is located—given it is visible within the field of view of any of the cameras. Therefore the network needs to decide which camera is responsible to keep track of a specific object at any time. Furthermore, the tracking camera has to decide when and to which camera it should transfer the tracking responsibility to. This process of transferring a tracking responsibility is know as *handover*. Each camera is self-aware, acts autonomously and tries to maximise its own utility. Nevertheless, every single camera is influenced and affected by the other cameras in the network and as a collective system all cameras share a common goal: keeping track of an object of interest with a certain utility. This utility is generated locally by each camera by tracking objects of interest. We consider the size as well as the confidence of the employed tracking algorithm as a common measurement of utility among all cameras. Inspired by market-mechanisms, tracking responsibilities are treated as goods and our cameras act as independent merchants able to buy and sell tracking responsibilities autonomously. This distributed handover process is depicted in Figure 7.7. This handover process has first been introduced by Esterle et. al [124]. Whenever a camera decides to sell an object, it initiates an auction by transmitting an object description to the other cameras in the network (cp. Figure 7(b)). Thereafter, the receiving cameras, generally willing to participate in such an auction, try to value this possible tracking responsibility (cp. Figure 7(c)). This valuation is based on an instantaneous utility of the corresponding object. To allow for a equivalent evaluation among all cameras, all cameras have to apply the same approach to calculate this instantaneous utility for the observed objects. The auctioneering cameras transfers the tracking responsibility at the end of the auction to the highest bidder (cp. Figure 7(d)). Instead of requesting the highest bidders valuation, only the second highest bid has to be paid. This Vickrey auction mechanism makes truthful bidding the dominant strategy for all participants and hence less vulnerable to malicious cameras. Additionally, this distributed approach allows continuous tracking of objects without any central component, analysing all data and coordinating tracking responsibilities.

This distributed handover approach gives rise to two important questions:

1. Which cameras should be invited for an auction in order to maximise the success of the auction while minimising the marketing effort?
2. At what time should the auctioneering camera try to sell the object of interest?

7.4.1.1 Building and Exploiting the Neighbourhood Relations

While cameras could offer their objects to all cameras in the network, only those cameras having a neighbouring field of view would actually have a chance to ‘see’ and hence valuate the object. To optimise their marketing efforts, cameras build up a knowledge about their neighbourhood by reasoning about the received bids after initiating an auction. Those neighbours having the object in their field of view

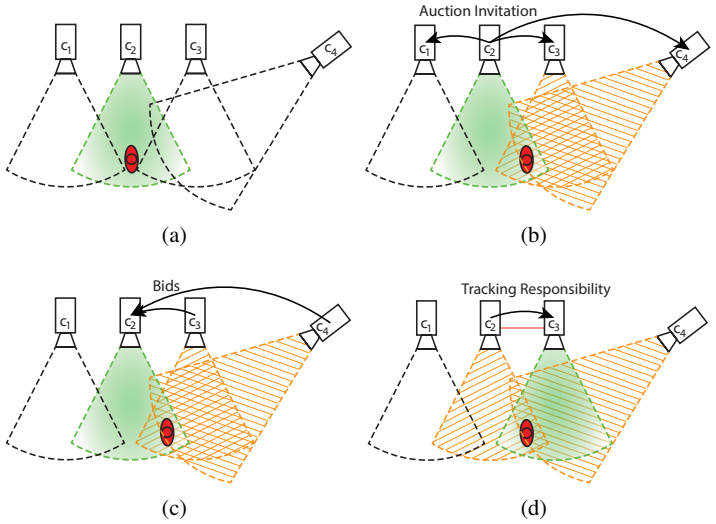

Fig. 7.7 Illustration of the market-based handover approach: an object of interest is tracked by camera c_2 in Illustration (a) indicated by the dashed green line and the shaded area representing the FOV. c_2 initiates an auction for an object as it is about to leave the FOV in Figure (b). Bids for taking over tracking responsibility are sent by camera c_3 and c_4 in Illustration (c) which have the object within their FOV as indicated by the orange dashed lines and the hatched area representing their FOV. c_3 wins the auction and the tracking responsibility is transferred from c_2 to c_3 in Illustration (d). A link in the vision graph is created (indicated as red line between c_2 and c_3).

will most likely have a field of view quite close to their own. As cameras might misidentify objects and hence mistakenly submit a bid with an valuation for the wrong object, but also due to possible changes in the network topology, we require a technique capable of forgetting about previously or incorrectly learnt neighbourhood relations. Inspired by the ant foraging process, we use artificial pheromones to build up a local neighbourhood graph on each camera. With every trade a selling camera i completes, it increases the strength of the link τ_{ix} to the respective neighbouring camera x by the value Δ. At the same time links evaporate continuously, ensuring cameras are able to *forget* now invalid information. The update rule for the artificial pheromones is shown in Equation 7.10.

$$\tau_{ix} = \begin{cases} (1-\rho)\cdot\tau_{ix} & \text{if no trade occurs on the edge} \\ (1-\rho)\cdot\tau_{ix}+\Delta & \text{if trade occurs on the edge} \end{cases} \tag{7.10}$$

As in ant colony optimisation, ρ is the evaporation rate parameter, which can be understood as a forgetting factor; higher values lead the pheromone to evaporate faster, enabling the system to adapt to changes quicker, but at a penalty of losing more historical vision graph information. However, our approach here is not ant colony optimisation, since pheromone information is not used to find optimal routes

through the network, but instead to represent a social environment of cameras with adjacent fields of view.

Learning about their local, social network, each camera is able to focus its marketing effort to those cameras with the highest probability of successfully participating in auctions. To ensure, new cameras can be included, cameras with a low or no link in the neighbourhood graph receive an auction invitation with a smaller probability.

As the vision graph is built up, the initial communication behaviour can be adapted. Specifically, when advertising an object that other cameras may wish to buy, a camera i sends a message to camera x with probability $P(i,x)$, otherwise it does not communicate with camera i at that time. We consider three different policies of determining these communication probabilities:

1. BROADCAST, which communicates with all available cameras in the network. This approach does not miss any camera but also generates a high overhead since it includes cameras which are not likely to respond.
2. STEP, in which an advertisement is sent to a camera if the strength of the link to that camera is above a certain threshold, otherwise the camera communicates with the other camera with a very low probability.
3. SMOOTH, in which the probability of communicating with another camera is based on the ratio between its link strength and that of the strongest link in its graph.

More formally, when employing a policy, the probability of camera i communicating with another camera x is given by

$$P_{\text{BROADCAST}}(i,x) = 1 \tag{7.11}$$

when using the BROADCAST policy. For STEP, the probability is given by

$$P_{\text{STEP}}(i,x) = \begin{cases} 1 & \text{if } (\tau_{ix} > \varepsilon) \vee (\tau_{im} = 0) \\ \eta & \text{otherwise} \end{cases} \tag{7.12}$$

where $\varepsilon = 0.1$ and m is the camera with the highest strength value, i.e.,

$$m = \underset{y}{\operatorname{argmax}}\, \tau_{iy}, \forall y.$$

For the SMOOTH policy, the communication probability is given by

$$P_{\text{SMOOTH}}(i,x) = \frac{1+\tau_{ix}}{1+\tau_{im}} \tag{7.13}$$

where m is again the camera with the highest strength value.

7.4.1.2 When to Handover

The timing of initiating auctions in a system observing an highly dynamic environment can be quite crucial. In a very straightforward approach, we enable our cameras to initiate auction in very short intervals, trying only to maximise the utility for each object without considering the possible processing overhead for other cameras in the network. We call this an ACTIVE auctioning schedule.

In contrast, cameras could only initiate auctions whenever the object is about to leave the field of view of a camera. In this approach, the camera would only try to sell the object when its own utility is very low. On one hand, the camera might have been better off selling the object earlier. On the other hand, this results in a possibly lower processing overhead for the other cameras in the network. This approach is called PASSIVE auctioning schedule.

Combining these two auction initiation schedules with the three previously discussed communication policies results in six different strategies. An operator, trying to balance the trade-off between communication overhead and tracking performance in the system, can select one of these six strategies. To start our distributed tracking application, objects of interest have to be defined. In our system, this basic mechanism is accomplished by a human operator who has to connect to a remote camera and select the object or person to be tracked in a user interface. This user interface is only required to initiate the tracking process and does not act as a central component or is not needed in any way besides initialisation, to support our approach.

In addition to the two proposed approaches, the cameras could also learn their individual timings for handover. Starting with an ACTIVE approach, each camera tries to sell an object in very short intervals. Whenever an object has been sold to a neighbouring camera, the selling camera can keep track of this information. As objects are unlikely to always use the exact same trajectory, the camera has to approximate the time of sending out an action invitation for each object based on its current trajectory and speed. This introduces an exploration-exploitation-dilemma in order to find the optimal timings. We propose to use a probability for exploration and reduce this value over time given the sale for the respective object is successful. With non-successful trades, the camera can assume a change in the network and increase the probability. This increase in exploration allows the camera to adapt faster to changed topology conditions.

7.4.1.3 Dynamic Strategy Selection

As discussed in the previous section, there are six different behavioural strategies available for our smart camera system. Each of this strategies gives rise to one out of two conflicting objectives: minimising network-wide communication or maximising system-wide tracking performance. In Lewis et al. [242] we compare the performance of a system where all cameras employ the same strategy and one where the cameras may use strategies differing from the other cameras in the network. A system is called to have a *homogeneous* configuration when all cameras apply the

same strategy. Alternatively, a camera network has a *heterogeneous* configuration when at least two cameras use differing strategies.

Permitting this heterogeneity in the network enables nodes to specialise to their local situation and hence allows for a wider range of global outcomes when compared to the homogeneous case. We speculate that optimal heterogeneous assignment of strategies can lead to the global performance of the network being strictly better in terms of both of the considered objectives. This would extend the previous Pareto efficient frontier with respect to the network-wide observed trade-off between communication overhead and achieved tracking performance. However, heterogeneity itself does not necessarily lead to better outcomes. It is also possible that nodes specialise wrongly, leading to a strictly worse global outcome when compared to any homogeneous case. Indeed, when considering all possible heterogeneous configurations for a given network, the number of possible configurations increases greatly compared to the homogeneous-only case. Having γ different strategies and n cameras in the network, an operator has to pick one configuration out of γ^n possible configurations.

The selection of an appropriate configuration, using a variety of strategies in the smart camera network, turns out to require knowledge of the camera setup, the environment as well as the movement patterns of the objects of interest. To neglect *a priori* knowledge of these parameters, we implemented multi-armed bandit problem solvers in every camera of the network [18]. This problem is analogous to being faced with n slot machine arms, where each pull of an arm returns a random reward drawn from an unknown distribution associated with that arm. Given m total arm pulls, the task is to select which arms to pull such that the total reward obtained is maximised. If the player were to know the distributions behind each arm, then the player could simply select the best arm for every pull. However, since the distributions are unknown, the player must sample from each arm in order to gain some knowledge of each arm's reward distribution. The multi-armed bandit problem therefore encapsulates another classic *exploration vs. exploitation* dilemma.

We consider three well-known bandit solvers from the literature: the simple EPSILON-GREEDY [400], UCB1, which is known to perform well in static problems [18], and SOFTMAX [382]. EPSILON-GREEDY requires an ε value to determine the amount of exploration. A low ε value ($\varepsilon \to 0$) results in random selection of algorithms while a high value ($\varepsilon \to 1$) selects greedily the best algorithm, based on the previous rewards. UCB1 requires no parameters. SOFTMAX uses a temperature value τ to steer the probability of selecting an arm based on the expected reward. This means, that high temperatures ($\tau \to 1$) result in a random selection where each arm has nearly the same probability while lower temperatures ($\tau \to 0$) tend to select the arm based purely on the expected reward.

While we aim to achieve global, network-wide objectives in terms of communication overhead and tracking utility, we only rely on corresponding local metrics in order to avoid exchange of information among the cameras. On one hand we use the number of messages sent by a camera at a given time step and denote this by *communication*. On the other hand, we have *utility* which is the sum of the obtained tracking performance over all objects tracked by this camera in the current time step,

plus its balance of payments from all trading activity during this time step. We use these two local measurements in a linear combination as a reward function for the bandit solvers on each camera:

$$reward = \alpha \times utility - (1-\alpha) \times communication. \tag{7.14}$$

Here the value α allows an network operator to balance the trade-off between required communication and achieved tracking utility.

The use of bandit solvers on the camera level allows the individual cameras to learn on their own which strategy fits them best, given a priority on either minimising communication or maximising tracking performance.

7.4.2 Benefits and Challenges at Levels of Self-Awareness

In this subsection we analyse our socially inspired learning techniques based on the different levels of computational self-awareness, targeting at the application explained in Section 7.4.1. A detailed discussion of the different levels of computational self-awareness is given in Chapter 4.

7.4.2.1 Stimulus-Awareness

In a stimulus aware system, a node has only knowledge of single stimuli. The node is not able to identify the source of a stimuli. Furthermore, does the node not have any knowledge about previous stimuli and hence can not infer possible future stimuli. For the discussed smart camera system, in a stimuli aware system only a single camera would be responsible for tracking objects as handover interactions would not be possible.

Benefits: No Benefits.

Drawbacks: A stimulus-aware smart camera system corresponds to a camera system with only a single camera. A tracking coordination is not possible.

7.4.2.2 Interaction-Awareness

An interaction-aware node knows that its actions trigger specific reactions from the local environment. In the smart camera system, ACTIVE and PASSIVE BROADCAST represent a simple interaction-aware system. Each camera triggers auction invitations based on the applied strategy expecting response from neighbouring cameras. Conversely, upon receiving such an invitation, a neighbouring camera will try to identify the object and submit a bid, again expecting the initiating camera to assign the tracking responsibility. Building up knowledge about its local communities, each camera could also facilitate an approach in order to focus marketing efforts. Nev-

ertheless, as an interaction-aware system does not possess knowledge about historical or future phenomena. Therefore, the system is not able to use an ant-inspired approach to build up the local neighbourhood graph nor to facilitate the STEP or SMOOTH communication policy.
Benefits: Possible coordination of tracking responsibilities.
Drawbacks: No focussed marketing efforts. No robustness towards changes in the topology. Only fixed strategy assignment.

7.4.2.3 Time-Awareness

With the introduction of time-awareness, a system gains knowledge about past and future events. In the smart camera system this relates to the ant-inspired neighbourhood graph and the evaporation of artificial pheromones allowing the network to 'forget' about changed environmental or social conditions. Only with time-awareness, a system can take full advantage of the proposed communication policies STEP and SMOOTH.
Benefits: Focussed marketing efforts possible when coordinating tracking responsibilities. Robustness towards changes in the network topology.
Drawbacks: Only fixed strategy assignments. No reasoning about current performance.

7.4.2.4 Goal-Awareness

Goal-awareness requires a node to possess knowledge about current goals and objectives but also about preferences and constraints. Having a local performance function combining the required communication effort and achieved tracking utility enables the nodes of our smart camera system to reason about their current performance.
Benefits: Reasoning about current performance allowing to optimize local behaviour.
Drawbacks: Reasoning possible but not changing of behaviour.

7.4.2.5 Meta-Self-Awareness

Having a node change its own level of computational self-awareness is considered meta-self-awareness. While our example of a smart camera system does not change these levels explicitly, we can reason about the performance of the different strategy representing different levels of self-awareness. The exploration of the different strategies and the dynamic selection of a specific strategy during runtime by means of a multi-armed bandit problem solver is an implementation of meta-self-awareness.

Benefits: Cameras can reason about the currently applied strategy and explore others possibly more beneficial.
Drawbacks: Cameras only observe their very own performance measurements and do not consider possible impacts on cameras in their neighbourhood.

7.4.3 Other Related Techniques

This section focuses on related techniques that we believe to be more relevant to self-awareness and self-expression, without meaning to be comprehensive.

Socially inspired learning in collective systems tries to enable individuals to learn about their social background and exploit this knowledge for their future actions and decisions. We focussed on the building up a social community on each individual node and the interactions within this community. While this allows exchange of information to a focussed group within the collective, it does not necessarily require consensus about certain states among all nodes of this group. As with dynamic environments, these individual social networks can change over time and strategies are needed to keep track of these changes. In our example, we induce our cameras with social behaviour by using market-mechanisms. Furthermore, we learn the social network of each individual camera online using an ant-inspired approach which also allows us to forget about these physical neighbourhoods again in case the environment changes. Nevertheless, the individual nodes do only learn from each other passively rather than actively exchanging knowledge. A probably obvious example for active knowledge exchange are consensus mechanisms where nodes exchange information to achieve a common knowledge base. Olfati-Saber [301] propose a distributed Kalman-filter [207] in order to exchange information and achieve a common knowledge base among nodes with different observation matrices. The approach allows to reduce disagreement of the estimates among nodes in a target tracking task.

An alternative has been introduced by Michalski [266] known as *inductive learning* technique which is also known as *transfer learning*. Here, an explicit teacher or even the environment provides examples, observations or facts about some phenomenon to nodes in the collective. The nodes make an inductive inference to achieve an accurate generalisation from multiple scattered facts and observations. Shaw and Sikora [360] introduce a distributed take on inductive learning. They break up the samples and use inductive learning on each single one of them in a distributed fashion. Using a genetic algorithm, they synthesise the results from each learning program into a final concept. Ontañón and Plaza [304] present an augmentation-based approach on multi-agent inductive learning in order to improve the individual learning capabilities of each node in the collective.

Stone [375] uses a layered learning approach in a multi-agent system. His reinforcement learning approach, called Team-Partitioned, Opaque-Transition Reinforcement Learning (TPOT-RL), deals with collaborating agents not necessarily able to observe the actions of another agent in the collective. By observing the long-term effects of their actions, agents can simultaneously learn collaborative policies.

In order to learn how multiple agents should coordinate their actions Claus and Boutilier [74] extend Q-learning to multi-agent setting. Here the Q-value function valuates the agents taking certain *joint actions* at a given state. Therefore, these learners are also known as *joint action learners*. In our smart camera example this would allow each social group to perform certain actions in a given situation in order to optimise the performance of the entire group. Transitively this would benefit the entire network. Nevertheless, joint action learners are incapable of applying mixed policies—where they select an action out of a set of available actions based on a certain distribution for a given situation.

References

1. Aberdeen, D., Baxter, J.: Emmerald: a fast matrix-matrix multiply using intel's SSE instructions. Concurrency and Computation: Practice and Experience **13**(2), 103–119 (2001)
2. Abramowitz, M., Stegun, I.: Handbook of Mathematical Functions. Dover Publications (1965)
3. Agarwal, A., Harrod, B.: Organic computing. Tech. Rep. White paper, MIT and DARPA (2006)
4. Agarwal, A., Miller, J., Eastep, J., Wentziaff, D., Kasture, H.: Self-aware computing. Tech. Rep. AFRL-RI-RS-TR-2009-161, MIT (2009)
5. Agne, A., Platzner, M., Lübbers, E.: Memory virtualization for multithreaded reconfigurable hardware. In: Proc. Int. Conf. on Field Programmable Logic and Applications (FPL), pp. 185–188. IEEE Computer Society (2011). DOI 10.1109/FPL.2011.42
6. Ahuja, S., Carriero, N., Gelernter, D.: Linda and friends. IEEE Computer **19**(8), 26–34 (1986). DOI 10.1109/MC.1986.1663305
7. Al-Naeem, T., Gorton, I., Babar, M.A., Rabhi, F., Benatallah, B.: A quality-driven systematic approach for architecting distributed software applications. In: Proceedings of the 27th International Conference on Software Engineering, ICSE '05, pp. 244–253. ACM, New York, NY, USA (2005). DOI 10.1145/1062455.1062508. URL `http://doi.acm.org/10.1145/1062455.1062508`
8. Ali, H.A., Desouky, A.I.E., Saleh, A.I.: Studying and Analysis of a Vertical Web Page Classifier Based on Continuous Learning Naive Bayes (CLNB) Algorithm, pp. 210–254. Information Science (2009)
9. Alippi, C., Boracchi, G., Roveri, M.: Just-in-time classifiers for recurrent concepts. IEEE Transactions on Neural Networks and Learning Systems **24**(4), 620–634 (2013)
10. Amir, E., Anderson, M.L., Chaudhri, V.K.: Report on DARPA workshop on self-aware computer systems. Tech. Rep. UIUCDCS-R-2007-2810, UIUC Comp. Sci. (2007)
11. ANA: Autonomic Network Architecture. URL `www.ana-project.org`. Website: `www.ana-project.org`, (accessed January 2015)
12. Angelov, P.: Nature-inspired methods for knowledge generation from data in real-time (2006). URL `http://www.nisis.risk-technologies.com/popup/Mallorca2006\_Papers/A333\_13774\_Nature-inspiredmethodsforKnowledgeGeneration\_Angelov.pdf`
13. Apache: Hadoop. `http://hadoop.apache.org/docs/r1.2.1/mapred\_tutorial.html`. Online; accessed 2-April-2015
14. Araya-Polo, M., Cabezas, J., Hanzich, M., Pericàs, M., Rubio, F., Gelado, I., Shafiq, M., Morancho, E., Navarro, N., Ayguadé, E., Cela, J.M., Valero, M.: Assessing accelerator-based HPC reverse time migration. IEEE Trans. Parallel Distrib. Syst. **22**(1), 147–162 (2011)

15. Asanovic, K., Bodik, R., Catanzaro, B.C., Gebis, J.J., Husbands, P., Keutzer, K., Patterson, D.A., Plishker, W.L., Shalf, J., Williams, S.W., Yelick, K.A.: The landscape of parallel computing research: A view from Berkeley. Tech. Rep. UCB/EECS-2006-183, EECS Department, University of California, Berkeley (2006)
16. Asendorpf, J.B., Warkentin, V., Baudonnière, P.M.: Self-awareness and other-awareness. ii: Mirror self-recognition, social contingency awareness, and synchronic imitation. Developmental Psychology **32**(2), 313 (1996)
17. Athan, T.W., Papalambros, P.Y.: A note on weighted criteria methods for compromise solutions in multi-objective optimization. Engineering Optimization **27**(2), 155–176 (1996)
18. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning **47**(2–3), 235–256 (2002)
19. Babaoglu, O., Binci, T., Jelasity, M., Montresor, A.: Firefly-inspired heartbeat synchronization in overlay networks. In: First International Conference on Self-Adaptive and Self-Organizing Systems (SASO), pp. 77–86 (2007)
20. Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. Pattern Analysis and Machine Intelligence **33**(8), 1619–1632 (2011)
21. Bader, J., Zitzler, E.: HypE: an algorithm for fast hypervolume-based many-objective optimization. Tech. Rep. TIK 286, Computer Engineering and Networks Laboratory, ETH Zurich, Zurich (2008)
22. Baena-García, M., Campo-Ávila, J.D., Fidalgo, R., Bifet, A.: Early drift detection method. In: Procedings of the 4th ECML PKDD International Workshop on Knowledge Discovery From Data Streams (IWKDDS), pp. 77–86. Berlin, Germany (2006)
23. Baker, S.: The identification of the self. Psyc. Rev. **4**(3), 272–284 (1897)
24. Banks, A., Gupta, R.: Mqtt version 3.1.1. `http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html` (2014)
25. Bartolini, D.B., Sironi, F., Maggio, M., Cattaneo, R., Sciuto, D., Santambrogio, M.D.: A Framework for Thermal and Performance Management. In: Workshop on Managing Systems Automatically and Dynamically (MAD) (2012)
26. Basheer, I.A., Hajmeer, M.: Artifcial neural networks: fundamentals, computing, design, and application. Journal of Microbiological Methods **43**(1), 3–31 (2000)
27. Basseur, M., Zitzler, E.: Handling uncertainty in indicator-based multiobjective optimization. International Journal of Computational Intelligence Research **2**(3), 255–272 (2006)
28. Basudhar, A., et al.: Constrained efficient global optimization with support vector machines. Struct. Multidiscip. Optim. **46**(2), 201—-221 (2012)
29. Baumann, A., Boltz, M., Ebling, J., Koenig, M., Loos, H.S., Marcel Merkel, W.N., Warzelhan, J.K., Yu, J.: A review and comparison of measures for automatic video surveillance systems. EURASIP Journal on Image and Video Processing **2008**(4) (2008). DOI 10.1155/2008/824726
30. Becker, T., Agne, A., Lewis, P.R., Bahsoon, R., Faniyi, F., Esterle, L., Keller, A., Chandra, A., Jensenius, A.R., Stilkerich, S.C.: EPiCS: Engineering proprioception in computing systems. In: Proc. Int. Conf. on Computational Science and Engineering (CSE), pp. 353–360. IEEE Computer Society (2012)
31. Ben-Hur, A., Weston, J.: A user's guide to support vector machines. Data Mining Techniques for the Life Sciences **609**, 223—-239 (2010)
32. Betts, A., Chong, N., Donaldson, A.F., Qadeer, S., Thompson, P.: GPUVerify: a verifier for GPU kernels. In: OOPSLA'12 (2012)
33. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: Multiobjective selection based on dominated hypervolume. European Journal on Operational Research **181**(3), 1653–1669 (2007)
34. Bevilacqua, F., Zamborlin, B., Sypniewski, A., Schnell, N., Guédy, F., Rasamimanana, N.: Continuous realtime gesture following and recognition. In: Gesture in embodied communication and human-computer interaction, pp. 73–84. Springer (2010)
35. Biehl, J.T., Adamczyk, P.D., Bailey, B.P.: Djogger: A mobile dynamic music device. In: CHI '06 Extended Abstracts on Human Factors in Computing Systems, pp. 556–561. ACM (2006)

36. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, United Kingdom (2005)
37. Bojic, I., Lipic, T., Podobnik, V.: Bio-inspired clustering and data diffusion in machine social networks. In: Computational Social Networks, pp. 51–79. Springer (2012)
38. Bongard, J., Lipson, H.: Evolved machines shed light on robustness and resilience. Proceedings of the IEEE **102**(5), 899–914 (2014)
39. Bongard, J., Zykov, V., Lipson, H.: Resilient machines through continuous self-modeling. Science **314**(5802), 1118–1121 (2006)
40. Borkar, S.: Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation. IEEE MICRO pp. 10–16 (2005)
41. Bouabene, G., Jelger, C., Tschudin, C., Schmid, S., Keller, A., May, M.: The Autonomic Network Architecture (ANA). Selected Areas in Communications, IEEE Journal on **28**(1), 4–14 (2010). DOI 10.1109/JSAC.2010.100102
42. Brdiczka, O., Crowley, J.L., Reignier, P.: Learning situation models in a smart home. IEEE Trans. Sys. Man Cyber. Part B **39**, 56–63 (2009)
43. Breiman, L.: Bagging predictors. Machine Learning **24**(2), 123–140 (1996)
44. Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)
45. Brockhoff, D., Zitzler, E.: Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods. In: Proceedings of the 2007 IEEE Congress on Evolutionary Computation, pp. 2086–2093 (2007)
46. Buchanan, J.T.: A naive approach for solving mcdm problems: The guess method. Journal of the Operational Research Society **48**(2), 202–206 (1997)
47. Buck, J.: Synchronous rhythmic flashing of fireflies. The Quarterly Review of Biology **13**(3), 301–314 (1938)
48. Buck, J.: Synchronous rhythmic flashing of fireflies II. Quarterly review of biology **63**(3), 265–289 (1988)
49. Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E., Qu, R.: Hyper-heuristics: A survey of the state of the art. Journal of the Operational Research Society **206**(1), 241–264 (2013)
50. Buschmann, F., Henney, K., Douglas, S.C.: Pattern-oriented software architecture: On patterns and pattern languages. John Wiley and Sons (2007)
51. Buss, A.H.: Self-consciousness and social anxiety. W. H. Freeman, San Fransisco, CA, USA (1980)
52. Calinescu, R., Ghezzi, C., Kwiatkowska, M., Mirandola, R.: Self-adaptive software needs quantitative verication at runtime. Communications of the ACM **55**(9), 69–77 (2012)
53. Caramiaux, B., Wanderley, M.M., Bevilacqua, F.: Segmenting and parsing instrumentalists' gestures. Journal of New Music Research **41**(1), 13–29 (2012)
54. Carver, C.S., Scheier, M.: Attention and Self-Regulation: A Control-Theory Approach to Human Behavior. Springer (1981)
55. Castro, L.N.d.: Fundamentals of natural computing: basic concepts, algorithms, and applications. Chapman & Hall/Crc Computer and Information Sciences (2006)
56. Chandra, A.: A methodical framework for engineering co-evolution for simulating socio-economic game playing agents. Ph.D. thesis, The University of Birmingham (2011)
57. Chandra, A., Nymoen, K., Volsund, A., Jensenius, A.R., Glette, K., Torresen, J.: Enabling participants to play rhythmic solos within a group via auctions. In: Proc. Int. Symp. on Computer Music Modeling and Retrieval (CMMR), pp. 674–689 (2012)
58. Chandra, A., Yao, X.: Ensemble learning using multi-objective evolutionary algorithms. Journal of Mathematical Modelling and Algorithms **5**(4), 417–445 (2006)
59. Chang, C., et al.: BEE2: a high-end reconfigurable computing system. IEEE Trans. Designs & Test of Computer (DT) **22**(2), 114–125 (2005)
60. Chen, J., John, L.K.: Efficient program scheduling for heterogeneous multi-core processors. In: Proc. Design Automation Conference (DAC). ACM (2009)
61. Chen, R., Lewis, P.R., Yao, X.: Temperature management for heterogeneous multi-core FPGAs using adaptive evolutionary multi-objective approaches. In: Proceedings of the International Conference on Evolvable Systems (ICES), pp. 101–108. IEEE (2014)

62. Chen, S., Langner, C.A., Mendoza-Denton, R.: When dispositional and role power fit: implications for self-expression and self-other congruence. Journal of Personality and Social Psychology **96**(3), 710–27 (2009)
63. Chen, T., Bahsoon, R.: Self-adaptive and sensitivity-aware qos modeling for the cloud. In: Proceedings of the 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS '13, pp. 43–52. IEEE Press, Piscataway, NJ, USA (2013). URL http://dl.acm.org/citation.cfm?id=2487336.2487346
64. Chen, T., Bahsoon, R.: Symbiotic and sensitivity-aware architecture for globally-optimal benefit in self-adaptive cloud. In: Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2014, pp. 85–94. ACM, New York, NY, USA (2014). DOI 10.1145/2593929.2593931. URL http://doi.acm.org/10.1145/2593929.2593931
65. Chen, T., Bahsoon, R., Yao, X.: Online qos modeling in the cloud: A hybrid and adaptive multi-learners approach. In: 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC), pp. 327–336. IEEE (2014)
66. Chen, T., Faniyi, F., Bahsoon, R., Lewis, P.R., Yao, X., Minku, L.L., Esterle, L.: The handbook of engineering self-aware and self-expressive systems. Tech. rep., EPiCS EU FP7 project consortium (2014). URL http://arxiv.org/abs/1409.1793. Available via EPiCS website and arXiv
67. Chen, X., Li, X., Wu, H., Qiu, T.: Real-time Object Tracking via CamShift-based Robust Framework. In: Int. Conf. on Information Science and Technology (ICIST). IEEE (2012)
68. Chow, G.C.T., Grigoras, P., Burovskiy, P., Luk, W.: An efficient sparse conjugate gradient solver using a beneš permutation network. In: 24th International Conference on Field Programmable Logic and Applications, FPL 2014, Munich, Germany, 2-4 September, 2014, pp. 1–7 (2014)
69. Chow, G.C.T., Tse, A.H.T., Jin, Q., Luk, W., Leong, P.H.W., Thomas, D.B.: A mixed precision monte carlo methodology for reconfigurable accelerator systems. In: Proceedings of the ACM/SIGDA 20th International Symposium on Field Programmable Gate Arrays, FPGA 2012, Monterey, California, USA, February 22-24, 2012, pp. 57–66 (2012)
70. Christensen, A.L., O'Grady, R., Dorigo, M.: From fireflies to fault-tolerant swarms of robots. IEEE Transactions on Evolutionary Computation **13**(4), 754–766 (2009)
71. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language (WSDL) 1.1. World Wide Web Consortium (2001)
72. Chu, F., Zaniolo, C.: Fast and light boosting for adaptive mining of data streams. In: Proceedings of the Eight Pacific-Asia Knowledge Discovery and Data Mining Conference (PAKDD), pp. 282–292. Sydney (2004)
73. Cichowski, A., Madden, C., Detmold, H., Dick, A., Van den Hengel, A., Hill, R.: Tracking Hand-off in Large Surveillance Networks. In: Proceedings of the International Conference Image and Vision Computing New Zealand, pp. 276–281. IEEE Computer Society Press (2009). DOI 10.1109/IVCNZ.2009.5378396
74. Claus, C., Boutilier, C.: The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems. In: Proceedings of the Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence, pp. 746–752. American Association for Artificial Intelligence, Menlo Park, CA, USA (1998)
75. Collins, N.: The analysis of generative music programs. Organised Sound **13**, 237–248 (2008)
76. Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. Pattern Analysis and Machine Intelligence **27**(10), 1631–1643 (2005). DOI 10.1109/tpami.2005.205
77. Colorni, A., Dorigo, M., Maniezzo, V., et al.: Distributed optimization by ant colonies. In: Proceedings of the first European conference on artificial life, vol. 142, pp. 134–142. Elsevier (1991)
78. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(5) (2003). DOI 10.1109/tpami.2003.1195991

79. Connors, K.: Chemical kinetics: the study of reaction rates in solution. VCH Publishers (1990)
80. Cox, M.: Metacognition in computation: A selected research review. Artificial Intelligence **169**(2), 104–141 (2005)
81. C.Pilato et al: Speeding-up expensive evaluations in highlevel synthesis using solution modeling and fitness inheritance, vol. 2, pp. 701—-723 (2010)
82. Cramer, T., Schmidl, D., Klemm, M., an Mey, D.: Openmp programming on intel xeon phi coprocessors: An early performance comparison. In: Many-core Applications Research Community (MARC) Symposium at RWTH Aachen University, November 29th-30th 2012, Aachen, Germany, pp. 38–44 (2012)
83. Crockford, D.: The application/json Media Type for JavaScript Object Notation (JSON). RFC 7159, RFC Editor (2014). URL `http://tools.ietf.org/pdf/rfc7159.pdf`
84. Czajkowski, T.S., Aydonat, U., Denisenko, D., Freeman, J., Kinsner, M., Neto, D., Wong, J., Yiannacouras, P., Singh, D.P.: From opencl to high-performance hardware on FPGAS. In: 22nd International Conference on Field Programmable Logic and Applications (FPL), Oslo, Norway, August 29-31, 2012, pp. 531–534 (2012)
85. Datta, K., et al.: Stencil computation optimization and auto-tuning on state-of-the-art multicore architectures. In: SC, p. 4. IEEE (2008)
86. Davidson, A.A., Owens, J.D.: Toward techniques for auto-tuning GPU algorithms. In: Applied Parallel and Scientific Computing - 10th International Conference, PARA 2010, Reykjavík, Iceland, June 6-9, 2010, Revised Selected Papers, Part II, pp. 110–119, Owner = Xinyu, Timestamp = 2015.05.07 (2010)
87. Day, J.: Patterns in Network Architecture: A Return to Fundamentals. Prentice Hall International (2008)
88. Day, J., Matta, I., Mattar, K.: Networking is IPC: A Guiding Principle to a Better Internet. In: Proceedings of the 2008 ACM CoNEXT Conference, CoNEXT '08, pp. 67:1–67:6. ACM, New York, NY, USA (2008). DOI 10.1145/1544012.1544079. URL `http://doi.acm.org/10.1145/1544012.1544079`
89. Dean, J., Ghemawat, S.: Mapreduce: Simplified data processing on large clusters. In: 6th Symposium on Operating System Design and Implementation (OSDI 2004), San Francisco, California, USA, December 6-8, 2004, pp. 137–150 (2004)
90. Deb, K.: Multi-objective optimization using evolutionary algorithms, vol. 16. John Wiley & Sons, England (2001)
91. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002)
92. Denholm, S., Inoue, H., Takenaka, T., Luk, W.: Application-specific customisation of market data feed arbitration. In: Proc. Int. Conf. on Field Programmable Technology (ICFPT), pp. 322–325. IEEE (2013)
93. Denholm, S., Inouey, H., Takenakay, T., Becker, T., Luk, W.: Low latency FPGA acceleration of market data feed arbitration. In: Proc. Int. Conf. on Application-Specific Systems, Architectures, and Processors (ASAP), pp. 36–40. IEEE (2014). DOI 10.1109/ASAP.2014.6868628
94. Dennett, D.C.: Consciousness Explained. Penguin Science (1993)
95. Dennis, J.B., Misunas, D.: A preliminary architecture for a basic data flow processor. In: Proceedings of the 2nd Annual Symposium on Computer Architecture, December 1974, pp. 126–132 (1974)
96. Dieber, B., Simonjan, J., Esterle, L., Rinner, B., Nebehay, G., Pflugfelder, R., Fernandez, G.J.: Ella: Middleware for multi-camera surveillance in heterogeneous visual sensor networks. In: Proc. Int. Conf. on Distributed Smart Cameras (ICDSC) (2013). DOI 10.1109/ICDSC.2013.6778223
97. Dieber, B., Simonjan, J., Esterle, L., Rinner, B., Nebehay, G., Pflugfelder, R., Fernandez, G.J.: Ella: Middleware for multi-camera surveillance in heterogeneous visual sensor networks. In: Proceedings of the Seventh International Conference on Distributed Smart Cameras (ICDSC), 2013, pp. 1–6. IEEE (2013)

98. Dietterich, T.G.: Ensemble methods in machine learning. In: Proceedings of the First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science, pp. 1–15. Springer-Verlag (2000)
99. Diguet, J.P., Eustache, Y., Gogniat, G.: Closed-loop–based Self-adaptive Hardware/Software-Embedded Systems: Design Methodology and Smart Cam Case Study. ACM Transactions on Embedded Computing Systems **10**(3), 1–28 (2011)
100. Dinh, M.N., Abramson, D., J. Chao, D.K., Gontarek, A., Moench, B., DeRose, L.: Debugging scientific applications with statistical assertions. Procedia Computer Science **9**(0), 1940––1949 (2012)
101. Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., Zambonelli, F.: A survey of autonomic communications. ACM Transactions on Autonomous and Adaptive Systems **1**(2), 223–259 (2006)
102. Dobson, S., Sterritt, R., Nixon, P., Hinchey, M.: Fulfilling the vision of autonomic computing. IEEE Computer **43**(1), 35 –41 (2010)
103. Dobzhansky, T., Hecht, M., Steere, W.: On some fundamental concepts of evolutionary biology. Evolutionary Biology **2**, 1–34 (1968)
104. Dorigo, M.: Optimization, learning and natural algorithms. Ph. D. Thesis, Politecnico di Milano, Italy (1992)
105. Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization. Computational Intelligence Magazine, IEEE **1**(4), 28–39 (2006)
106. Dorigo, M., Blum, C.: Ant colony optimization theory: A survey. Theoretical computer science **344**(2), 243–278 (2005)
107. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **26**(1), 29–41 (1996)
108. Dutta, R., Rouskas, G., Baldine, I., Bragg, A., Stevenson, D.: The SILO Architecture for Services Integration, controL, and Optimization for the Future Internet. In: IEEE International Conference onCommunications (ICC), pp. 1899–1904 (2007). DOI 10.1109/ICC.2007.316
109. Duval, S., Wicklund, R.A.: A theory of objective self awareness. Academic Press (1972)
110. Eckart Zitzler, S.K.: Indicator-based selection in multiobjective search. In: Proceedings of Parallel Problem Solving from Nature (PPSN), vol. 3242, pp. 832–842 (2004)
111. Ehrgott, M.: Other methods for pareto optimality. In: Multicriteria Optimization, Lecture Notes in Economics and Mathematical Systems, vol. 491, pp. 77–102. Springer (2000)
112. Eiben, A.E., Smith, J.E.: Introduction to evolutionary computing. Springer (2003)
113. Eigenfeldt, A., Pasquier, P.: Considering vertical and horizontal context in corpus-based generative electronic dance music. In: Proceedings of the Fourth International Conference on Computational Creativity, p. 72 (2013)
114. Eigenfeldt, A., Pasquier, P.: Evolving structures for electronic dance music. In: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, GECCO '13, pp. 319–326. ACM, New York, NY, USA (2013)
115. Elkhodary, A., Esfahani, N., Malek, S.: Fusion: a framework for engineering self-tuning self-adaptive software systems. In: Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering, FSE '10, pp. 7–16. ACM, New York, NY, USA (2010). DOI 10.1145/1882291.1882296. URL `http://doi.acm.org/10.1145/1882291.1882296`
116. Elliott, G.T., Tomlinson, B.: Personalsoundtrack: context-aware playlists that adapt to user pace. In: CHI'06 extended abstracts on Human factors in computing systems, pp. 736–741. ACM (2006)
117. Ellis, T., Makris, D., Black, J.: Learning a Multi-camera Topology. In: Proceedings of the Joint International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, pp. 165–171. IEEE Computer Society Press (2003)
118. Elwell, R., Polikar, R.: Incremental learning of concept drift in nonstationary environments. IEEE Transactions on Neural Networks **22**, 1517–1531 (2011)

119. Endo, T., Matsuoka, S.: Massive supercomputing coping with heterogeneity of modern accelerators. In: 22nd IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2008, Miami, Florida USA, April 14-18, 2008, pp. 1–10 (2008)
120. Erdem, U.M., Sclaroff, S.: Look there! predicting where to look for motion in an active camera network. In: Proceedings of the IEEE Conference on Vision and Signal-based Surveillance, pp. 105–110. Como, Italy (2005)
121. Esterle, L., Lewis, P.R., Bogdanski, M., Rinner, B., Yao, X.: A socio-economic approach to online vision graph generation and handover in distributed smart camera networks. In: Proc. Int. Conf. on Distributed Smart Cameras (ICDSC), pp. 1–6. IEEE (2011). DOI 10.1109/ICDSC.2011.6042902
122. Esterle, L., Lewis, P.R., Caine, H., Yao, X., Rinner, B.: CamSim: A distributed smart camera network simulator. In: Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems Workshops, pp. 19–20. IEEE Computer Society Press (2013). DOI 10.1109/SASOW.2013.11
123. Esterle, L., Lewis, P.R., Rinner, B., Yao, X.: Improved adaptivity and robustness in decentralised multi-camera networks. In: Proceedings of the International Conference on Distributed Smart Cameras, pp. 1–6. ACM (2012)
124. Esterle, L., Lewis, P.R., Yao, X., Rinner, B.: Socio-economic vision graph generation and handover in distributed smart camera networks. ACM Transactions on Sensor Networks **10**(2), 20:1–20:24 (2014). DOI 10.1145/2530001
125. Faniyi, F., Lewis, P.R., Bahsoon, R., Xao, X.: Architecting self-aware software systems. In: Proc. IEEE/IFIP Conf. on Software Architecture (WICSA), pp. 91–94. IEEE (2014)
126. Farrell, R., Davis, L.S.: Decentralized discovery of camera network topology. In: Proceedings of the International Conference on Distributed Smart Cameras, pp. 1–10. IEEE Computer Society Press (2008). DOI 10.1109/ICDSC.2008.4635696
127. Fels, S., Hinton, G.: Glove-talk: A neural network interface between a data-glove and a speech synthesizer. IEEE Trans. Neural Networks **4**(1), 2–8 (1993)
128. Feng, W.: Making a case for efficient supercomputing. ACM Queue **1, Owner = Xinyu, Timestamp = 2015.05.07**(7), 54–64 (2003)
129. Fenigstein, A., Scheier, M.F., Buss, A.H.: Public and private self-consciousness: Assessment and theory. Journal of Consulting and Clinical Psychology **43**(4), 522–527 (1975)
130. Fern, A., Givan, R.: Online ensemble learning: An empirical study. Machine Learning **53**(1–2), 71–109 (2003)
131. Fette, B.: Cognitive radio technology. Academic Press (2009)
132. Fiebrink, R., Trueman, D., Cook, P.R.: A meta-instrument for interactive, on-the-fly machine learning. In: Proceedings of the International Conference on New Interfaces for Musical Expression. Pittsburgh (2009)
133. Fielding, R.T., Taylor, R.N.: Principled design of the modern web architecture. ACM Trans. Internet Technol. **2**(2), 115–150 (2002). DOI 10.1145/514183.514185. URL `http://doi.acm.org/10.1145/514183.514185`
134. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the 13th International Conference on Machine Learning, pp. 148–156 (1996)
135. Froming, W.J., Walker, G.R., Lopyan, K.J.: Public and private self-awareness: When personal attitudes conflict with societal expectations. Journal of Experimental Social Psychology **18**(5), 476 – 487 (1982). DOI DOI:10.1016/0022-1031(82)90067-1
136. Fu, H., Sendhoff, B., Tang, K., Yao, X.: Finding robust solutions to dynamic optimization problems. In: Proceedings of the 16th European conference on Applications of Evolutionary Computation (EvoApplications), pp. 616–625 (2013)
137. Funie, A., Salmon, M., Luk, W.: A hybrid genetic-programming swarm-optimisation approach for examining the nature and stability of high frequency trading strategies. In: 13th International Conference on Machine Learning and Applications ICMLA, pp. 29–34. Detroit,USA (2014). DOI 10.1109/ICMLA.2014.11. URL `http://dx.doi.org/10.1109/ICMLA.2014.11`
138. Gallup, G.G.: Chimpanzees: self-recognition. Science (1970)

139. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Proceedings of the 7th Brazilian Symposium on Artificial Intelligence (SBIA) - Lecture Notes in Computer Science, vol. 3171, pp. 286–295. Springer, São Luiz do Maranhão, Brazil (2004)
140. Gao, J., Fan, W., Han, J.: On appropriate assumptions to mine data streams: Analysis and practice. In: Seventh IEEE International Conference on Data Mining (ICDM), pp. 143–152 (2007)
141. Garlan, D., Cheng, S.W., Huang, A.C., Schmerl, B., Steenkiste, P.: Rainbow: architecture-based self-adaptation with reusable infrastructure. Computer **37**(10), 46–54 (2004)
142. Gelenbe, E., Loukas, G.: A self-aware approach to denial of service defence. Computer Networks **51**, 1299—1314 (2007)
143. Goto, M.: Active music listening interfaces based on signal processing. In: IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 4, pp. 1441—1444 (2007)
144. Gouin-Vallerand, C., Abdulrazak, B., Giroux, S., Mokhtari, M.: Toward autonomic pervasive computing. In: Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services, iiWAS '08, pp. 673–676. ACM, New York, NY, USA (2008)
145. Goukens, C., Dewitte, S., Warlop, L.: Me, myself, and my choices: The influence of private self-awareness on preference-behavior consistency. Tech. rep., Katholieke Universiteit Leuven (2007)
146. Grabner, H., Bischof, H.: On-line boosting and vision. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 260–267 (2006)
147. Grabner, H., Leistner, C., Bischof, H.: Semi-supervised On-Line boosting for robust tracking. In: European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 5302, pp. 234–247 (2008)
148. Group, K.: The opencl specification, version: 1.1. `http://www.khronos.org/registry/cl/specs/opencl-1.1.pdf`. [Online; accessed 2-April-2015]
149. Gudger, E.W.: A historical note on the synchronous flashing of fireflies. Science **50**(1286), 188–190 (1919)
150. Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.J., Nielsen, H.F., Karmarkar, A., Lafon, Y.: SOAP Version 1.2. World Wide Web Consortium (2007)
151. Guo, C., Luk, W.: Accelerating maximum likelihood estimation for hawkes point processes. In: Proc. Int. Conf. on Field Programmable Logic and Applications (FPL), pp. 1–6. IEEE (2013)
152. Guo, C., Luk, W.: Accelerating parameter estimation for multivariate self-exciting point processes. In: Proc. Int. Symp. on Field-Programmable Gate Arrays (FPGA), pp. 181–184. ACM (2014). DOI 10.1145/2554688.2554765
153. Haikonen, P.O.: Reflections of consciousness: The mirror test. In: Proceedings of the 2007 AAAI Fall Symposium on Consciousness and Artificial Intelligence, pp. 67–71 (2007)
154. Hamid, R., Maddi, S., Johnson, A., Bobick, A., Essa, I., Isbell, C.: A novel sequence representation for unsupervised analysis of human activities. Artificial Intelligence **173**(14), 1221 – 1244 (2009). DOI DOI:10.1016/j.artint.2009.05.002
155. Hansen, N.: The CMA evolution strategy: A comparing review. In: J. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea (eds.) Towards a New Evolutionary Computation, *Studies in Fuzziness and Soft Computing*, vol. 192, pp. 75–102. Springer Berlin Heidelberg (2006)
156. Hansen, N.: The CMA evolution strategy: A comparing review. In: J. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea (eds.) Towards a New Evolutionary Computation, *Studies in Fuzziness and Soft Computing*, vol. 192, pp. 75–102. Springer Berlin Heidelberg (2006)
157. Happe, M., Agne, A., Plessl, C.: Measuring and predicting temperature distributions on FPGAs at run-time. In: Proc. Int. Conf. on ReConFigurable Computing and FPGAs (ReConFig), pp. 55–60. IEEE Computer Society, Los Alamitos, CA, USA (2011). DOI 10.1109/ReConFig.2011.59
158. Happe, M., Hangmann, H., Agne, A., Plessl, C.: Eight ways to put your FPGA on fire – a systematic study of heat generators. In: Proc. Int. Conf. on ReConFigurable Computing and FPGAs (ReConFig). IEEE Computer Society (2012). Received Best Paper Award

159. Happe, M., Huang, Y., Keller, A.: Dynamic protocol stacks in smart camera networks. In: Proc. Int. Conf. on ReConFigurable Computing and FPGAs (ReConFig). IEEE (2014). To appear
160. Happe, M., Traber, A., Keller, A.: Preemptive Hardware Multitasking in ReconOS. In: International Symposium on Applied Reconfigurable Computing (ARC), Springer (2015)
161. Hart, J.W., Scassellati, B.: Robotic self-modeling. In: J. Pitt (ed.) The Computer After Me, pp. 207–218. Imperial College Press / World Scientific Book (2014)
162. Heath, D., Jarrow, R., Morton, A.: Bond pricing and the term structure of interest rates: A new methodology for contingent claims valuation. Econometrica **60**(77-105), 60–65 (1992)
163. Hernandez, H., Blum, C.: Distributed graph coloring in wireless ad hoc networks: A lightweight algorithm based on japanese tree frogs' calling behaviour. In: Proceedings of the 4th Joint IFIP Wireless and Mobile Networking Conference (WMNC), pp. 1–7 (2011)
164. Herzen, B.V.: Signal processing at 250 mhz using high-performance FPGA's. In: Proceedings of the 1997 ACM Fifth International Symposium on Field-programmable Gate Arrays, pp. 62–68 (1997)
165. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(8), 832–844 (1998)
166. Ho, T.K., Hull, J.J., Srihari, S.N.: Decision combination in multiple classifier systems. IEEE Transactions on Pattern Analysis and Machine Intelligence **16**(1), 66–75 (1994)
167. Ho, T.S.Y., b. Lee, S.: Term structure movements and pricing interest rate contingent claims. Journal of Finance **41**(5), 1986 (1011–1029)
168. Hoare, C.A.R.: An axiomatic basis for computer programming. Commun. ACM **12**(10), 576–580 (1969)
169. Hockman, J.A., Wanderley, M.M., Fujinaga, I.: Real-time phase vocoder manipulation by runner's pace. In: Proceedings of the International Conference on New Interfaces for Musical Expression (2009)
170. Hoffmann, H., Eastep, J., Santambrogio, M., Miller, J., Agarwal, A.: Application heartbeats for software performance and health. In: ACM SIGPLAN Notices, vol. 45, pp. 347–348. ACM (2010)
171. Hoffmann, H., Eastep, J., Santambrogio, M.D., Miller, J.E., Agarwal, A.: Application Heartbeats: A Generic Interface for Specifying Program Performance and Goals in Autonomous Computing Environments. In: International Conference on Autonomic Computing (ICAC) (2010)
172. Hoffmann, H., Holt, J., Kurian, G., Lau, E., Maggio, M., Miller, J.E., Neuman, S.M., Sinangil, M., Sinangil, Y., Agarwal, A., Chandrakasan, A.P., Devadas, S.: Self-aware computing in the angstrom processor. In: Proceedings of the 49th Annual Design Automation Conference, DAC '12, pp. 259–264. ACM, New York, NY, USA (2012)
173. Hoffmann, H., Maggio, M., Santambrogio, M.D., Leva, A., Agarwal, A.: SEEC: A general and extensible framework for self-aware computing. Tech. Rep. MIT-CSAIL-TR-2011-046, Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology (2011)
174. Holland, B., et al.: An analytical model for multilevel performance prediction of Multi-FPGA systems. ACM Trans. Reconfigurable Technol. Syst **4**(3), 27–28 (2011)
175. Holland, O., Goodman, R.B.: Robots with internal models: A route to machine consciousness? Journal of Consciousness Studies **10**(4), 77–109 (2003)
176. Holopainen, R.: Self-organised sound with autonomous instruments: Aesthetics and experiments. Ph.D. thesis, University of Oslo (2012)
177. Hölzl, M., Wirsing, M.: Towards a system model for ensembles. In: Formal Modeling: Actors, Open Systems, Biological Systems, pp. 241–261. Springer (2011)
178. Hölzl, M., Wirsing, M.: Issues in engineering self-aware and self-expressive ensembles. In: J. Pitt (ed.) The Computer After Me, pp. 37–54. Imperial College Press / World Scientific Book (2014)
179. Horn, J., Nafpliotis, N., Goldberg, D.E.: A niched pareto genetic algorithm for multiobjective optimization. In: Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, pp. 82–87. IEEE (1994)

180. Horn, P.: Autonomic computing: IBM's perspective on the state of information technology. Armonk, NY, USA. International Business Machines Corporation. (2001)
181. Hosseini, M.J., Ahmadi, Z., Beigy, H.: Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification. Evolving Systems **4**(1), 43–60 (2013)
182. hsing Hsu, C., chun Feng, W.: Reducing overheating-induced failures via performance-aware cpu power management. In: In The 6th International Conference on Linux Clusters: The HPC Revolution (2005)
183. Hu, F., Evans, J.J.: Power and environment aware control of beowulf clusters. Cluster Computing **12**, 299–308 (2009)
184. Hu, W., Tan, T., Wang, L., Maybank, S.: A Survey on Visual Surveillance of Object Motion and Behaviors. IEEE Transactions on Systems, Man and Cybernetics, Part C **34**(3), 334–352 (2004)
185. Huang, T., Russell, S.: Object Identification in a Bayesian Context. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1276–1283 (1997)
186. Huebscher, M., McCann, J.: Simulation model for self-adaptive applications in pervasive computing. In: Proceedings of the 15th International Workshop on Database and Expert Systems Applications,, pp. 694–698. IEEE Computer Society (2004)
187. Hume, D.: A Treatise of Human Nature. Gutenberg eBook (1739). Digital edition, 2010, available at http://www.gutenberg.org/ebooks/4705, retrieved 21st January 2013
188. Hunkeler, U., Truong, H.L., Stanford-Clark, A.: Mqtt-s—a publish/subscribe protocol for wireless sensor networks. In: In Proceedings of the Third International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE), 2008, pp. 791–798. IEEE (2008)
189. Hunt, A., Wanderley, M.M., Paradis, M.: The importance of parameter mapping in electronic instrument design. In: Proceedings of the International Conference on New Interfaces for Musical Expression, pp. 1–6. National University of Singapore, Singapore, Singapore (2002)
190. IBM: An architectural blueprint for autonomic computing (2003)
191. Iglesia, D.: Mobmuplat (iOS application). Iglesia Intermedia (2013)
192. Inc., I.: Sophisticated library for vector parallelism. `http://software.intel.com/en-us/articles/intel-array-building-blocks/`. [Online; accessed 2-April-2015]
193. Investigating RINA as an Alternative to TCP/IP. URL `irati.eu`. Website: `irati.eu`, (accessed January 2015)
194. Ishibuchi, H., Murata, T.: A multiobjective genetic local search algorithm and its application to flowshop scheduling. IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews **28**(3), 392–403 (1998)
195. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Iterative approach to indicator-based multiobjective optimization. In: Proceedings of the 2007 IEEE Congress on Evolutionary Computation, pp. 3967–3974. IEEE (2007)
196. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization. In: Proceedings of the 3rd International Workshop on Genetic and Evolving Systems (GEFS), pp. 47–52. IEEE (2008)
197. J., C., G., S., D., G.A.: High-level synthesis of in-circuit assertions for verification, debugging, and timing analysis. International Journal of Reconfigurable Computing **2011** (2011)
198. James, W.: The principles of psychology. Henry Holt & Co. (1890)
199. Janusevskis, J., et al.: Expected improvements for the asynchronous parallel global optimization of expensive functions: Potentials and challenges. In: Learning and Intelligent Optimization, pp. 413—-418. Springer (2012)
200. Javed, O., Khan, S., Rasheed, Z., Shah, M.: Camera Handoff: Tracking in Multiple Uncalibrated Stationary Cameras. In: Proceedings of the Workshop on Human Motion, pp. 113–118. IEEE Computer Society Press (2000). DOI 10.1109/HUMO.2000.897380
201. Javed, O., Rasheed, Z., Shafique, K., Shah, M.: Tracking across Multiple Cameras Disjoint Views. In: Proceedings of IEEE International Conference on Computer Vision, p. 952–957 (2003)

202. Jia, J., Veeravalli, B., Ghose, D.: Adaptive load distribution strategies for divisible load processing on resource unaware multilevel tree networks. IEEE Transactions on Computers **56**(7), 999–1005 (2007)
203. Jin, Q., et al.: Optimising explicit finite difference option pricing for dynamic constant reconfiguration. In: Proc. FPL, pp. 165—-172 (2012)
204. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. of Global Optimization **13**(4), 455—-492 (1998)
205. Jones, P., Cho, Y., Lockwood, J.: Dynamically optimizing FPGA applications by monitoring temperature and workloads. In: Proc. Int. Conf. on VLSI Design (VLSID). IEEE (2007)
206. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-Learning-detection. Pattern Analysis and Machine Intelligence **34**(7), 1409–1422 (2012)
207. Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. Journal of Fluids Engineering **82**(1), 35–45 (1960)
208. Kamil, S., Chan, C., Oliker, L., Shalf, J., Williams, S.: An auto-tuning framework for parallel multicore stencil computations. In: Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on, pp. 1–12. IEEE Computer Society Press (2010)
209. Kang, J., Cohen, I., Medioni, G.: Continuous Tracking within and across Cameras Streams. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 267–272 (2003)
210. Kant, I.: The critique of pure reason. Gutenberg eBook (1781). Digital edition, 2003, available at http://www.gutenberg.org/ebooks/4280, retrieved 21st January 2013
211. Kela, J., Korpipää, P., Mäntyjärvi, J., Kallio, S., Savino, G., Jozzo, L., Marca, D.: Accelerometer-based gesture control for a design environment. Personal and Ubiquitous Computing **10**(5), 285–299 (2006)
212. Keller, A., Borkmann, D., Neuhaus, S., Happe, M.: Self-awareness in computer networks. Int. Journal of Reconfigurable Computing (IJRC) (2014). DOI 10.1155/2014/692076
213. Keller, A., Plattner, B., Lübbers, E., Platzner, M., Plessl, C.: Reconfigurable nodes for future networks. In: Proc. IEEE Globecom Workshop on Network of the Future (FutureNet), pp. 372–376. IEEE (2010). DOI 10.1109/GLOCOMW.2010.5700341
214. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. IEEE Computer **36**(1), 41–50 (2003)
215. Kettnaker, V., Zabith, R.: Bayesian Multi- camera Surveillance. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition, pp. 117–123 (1999)
216. Khan, M.I., Rinner, B.: Energy-aware task scheduling in wireless sensor networks based on cooperative reinforcement learning. In: Proceedings of the International Conference on Communications Workshops (ICCW). IEEE (2014). DOI 10.1109/ICCW.2014.6881310
217. Khare, V., Yao, X., Deb, K.: Performance scaling of multi-objective evolutionary algorithms. In: Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol. 2632, pp. 376–390. Springer (2003)
218. Kim, H.S., Sherman, D.K.: Express yourself: Culture and the effect of self-expression on choice. Journal of Personality and Social Psychology **92**(1), 1–11 (2007). DOI DOI:10.1037/0022-3514.92.1.1
219. Kim, J., Seo, S., Lee, J., Nah, J., Jo, G., Lee, J.: Opencl as a unified programming model for heterogeneous CPU/GPU clusters. In: Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPOPP 2012, New Orleans, LA, USA, February 25-29, 2012, pp. 299–300 (2012)
220. Kittler, J., Hatef, M., Duin, R.P., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(3), 226–239 (1998)
221. Klinglmayr, J., Bettstetter, C.: Self-organizing synchronization with inhibitory-coupled oscillators: Convergence and robustness. ACM Transactions on Autonomous and Adaptive Systems (TAAS) **7**(3), 30:1–30:22 (2012)
222. Klinglmayr, J., Kirst, C., Bettstetter, C., Timme, M.: Guaranteeing global synchronization in networks with stochastic interactions. New Journal of Physics **14**(7), 073,031:1–073,031:13 (2012)

223. Knutzen, H., Nymoen, K., Torresen, J.: PheroMusic [iOS application]. URL \url{http://itunes.apple.com/app/pheromusic/id910100415}
224. Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: An ensemble method for drifting concepts. Journal of Machine Learning Research **8**, 2755–2790 (2007)
225. Koski, J., Silvennoinen, R.: Norm methods and partial weighting in multicriterion optimization of structures. International Journal for Numerical Methods in Engineering **24**(6), 1101–1121 (1987)
226. Kramer, J., Magee, J.: Self-managed systems: an architectural challenge. In: Future of Software Engineering, 2007. FOSE'07, pp. 259–268. IEEE (2007)
227. Krishnamoorthy, S., et al.: Effective automatic parallelization of stencil computations. In: ACM Sigplan Notices, vol. 42, pp. 235–244. ACM (2007)
228. Kuhn, H.W., Yaw, B.: The Hungarian Method for the Assignment Problem. Naval Research Logistics Quarterly pp. 83–97 (1955)
229. Kuncheva, L.I.: A theoretical study on six classifier fusion strategies. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(2), 281–286 (2002)
230. Kurek, M., Becker, T., Chau, T.C., Luk, W.: Automating optimization of reconfigurable designs. In: Proc. Int. Symp. on Field-Programmable Custom Computing Machines (FCCM), pp. 210–213. IEEE (2014). DOI 10.1109/FCCM.2014.65
231. Kurek, M., Becker, T., Luk, W.: "parametric optimization of reconfigurable designs using machine learning. In: ARC'13, pp. 134—-145 (2013)
232. Legrain, L., Cleeremans, A., Destrebecqz, A.: Distinguishing three levels in explicit self-awareness. Consciousness and Cognition **20**, 578–585 (2011)
233. Legrand, D.: Pre-reflective self-as-subject from experiential and empirical perspectives. Consciousness and Cognition **16**(3), 583–599 (2007)
234. Leidenfrost, R., Elmenreich, W.: Firefly clock synchronization in an 802.15. 4 wireless network. EURASIP Journal on Embedded Systems **2009**, 7:1–7:17 (2009)
235. Leland, W., Taqqu, M., Willinger, W., Wilson, D.: On the self-similar nature of ethernet traffic (extended version). Networking, IEEE/ACM Transactions on **2**(1), 1–15 (1994). DOI 10.1109/90.282603
236. Leutenegger, S., Chli, M., Siegwart, R.Y.: BRISK: Binary robust invariant scalable keypoints. In: Proceedings of the International Conference on Computer Vision, pp. 2548–2555. IEEE (2011). DOI 10.1109/iccv.2011.6126542
237. Lewis, P., Platzner, M., Yao, X.: An outlook for self-awareness in computing systems. Awareness Magazine (2012). DOI 10.2417/3201203.004093
238. Lewis, P.R., Chandra, A., Faniyi, F., Glette, K., Chen, T., Bahsoon, R., Torresen, J., Yao, X.: Architectural aspects of self-aware and self-expressive computing systems. Computer **48**(8) (2015)
239. Lewis, P.R., Chandra, A., Parsons, S., Robinson, E., Glette, K., Bahsoon, R., Torresen, J., Yao, X.: A Survey of Self-Awareness and Its Application in Computing Systems. In: Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems Workshops (2011)
240. Lewis, P.R., Chandra, A., Parsons, S., Robinson, E., Glette, K., Bahsoon, R., Torresen, J., Yao, X.: A survey of self-awareness and its application in computing systems. In: Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems Workshops, pp. 102–107. IEEE Computer Society, Ann Arbor, MI, USA (2011)
241. Lewis, P.R., Esterle, L., Chandra, A., Rinner, B., Torresen, J., Yao, X.: Static, dynamic, and adaptive heterogeneity in distributed smart camera networks. ACM Trans. Auton. Adapt. Syst. **10**(2), 8:1–8:30 (2015). DOI 10.1145/2764460
242. Lewis, P.R., Esterle, L., Chandra, A., Rinner, B., Yao, X.: Learning to be different: Heterogeneity and efficiency in distributed smart camera networks. In: Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems (SASO), pp. 209–218. IEEE Computer Society Press (2013). DOI 10.1109/SASO.2013.20
243. Lewis, P.R., Marrow, P., Yao, X.: Resource allocation in decentralised computational systems: An evolutionary market based approach. Autonomous Agents and Multi-Agent Systems **21**(2), 143–171 (2010)

244. Li, B., Li, J., Tang, K., Yao, X.: An improved two archive algorithm for many-objective optimization. In: Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 2869–2876. IEEE (2014)
245. Li, G., Gopalakrishnan, G.: Scaleable SMT-based verification of GPU kernel functions. In: FSE-18 (2010)
246. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. IEEE Transactions on Evolutionary Computation **13**(2), 284–302 (2009)
247. Li, Y., Bhanu, B.: Utility-based Camera Assignment in a Video Network: A Game Theoretic Framework. Sensors Journal **11**(3), 676–687 (2011)
248. Liang, C.J.M., Liu, J., Luo, L., Terzis, A., Zhao, F.: Racnet: A high-fidelity data center sensing network. Proc. SenSys (2009, Owner = Xinyu, Timestamp = 2015.05.07)
249. Liu, J., Zhong, L., Wickramasuriya, J., Vasudevan, V.: uwave: Accelerometer-based personalized gesture recognition and its applications. Pervasive and Mobile Computing **5**(6), 657–675 (2009)
250. Liu, Y., Yao, X.: Ensemble learning via negative correlation. Neural Networks **12**(10), 1399–1404 (1999)
251. Lübbers, E., Platzner, M.: ReconOS: Multithreaded programming for reconfigurable computers. ACM Transactions on Embedded Computing Systems (TECS) **9** (2009)
252. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 674–679 (1981)
253. Lübbers, E., Platzner, M.: Cooperative multithreading in dynamically reconfigurable systems. In: Proc. Int. Conf. on Field Programmable Logic and Applications (FPL), pp. 1–4. IEEE (2009)
254. Makris, D., Ellis, T., Black, J.: Bridging the Gaps between Cameras. In: Proceedings of Conference on Computer Vision and Pattern Recognition, vol. 2 (2004)
255. Marler, R.T., Arora, J.S.: Function-transformation methods for multi-objective optimization. Engineering Optimization **37**(6), 551–570 (2005)
256. Marrow, P.: Nature-inspired computing technology and applications. BT Technology Journal **18**(4), 13–23 (2000)
257. Marsaglia, G., Bray, T.A.: A convenient method for generating normal variables. SIAM Review **6**(3), 260–264 (1964)
258. Masahiro, N., Takaesu, H., Demachi, H., Oono, M., Saito, H.: Development of an automatic music selection system based on runner's step frequency. In: Proc. of 2008 International Conf. on Music Information Retrieval, pp. 193–8 (2008)
259. Massie, M.L., b, B.N.C., Culler, D.E.: The Ganglia distributed monitoring system: design, implementation, and experience. Parallel Computing **30**, 817–840 (2004)
260. Mathar, R., Mattfeldt, J.: Pulse-coupled decentral synchronization. SIAM Journal on Applied Mathematics **56**(4), 1094–1106 (1996)
261. Max [computer software]. URL `\url{http://cycling74.com}`
262. Mckay, M.D., et al.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics pp. 55—-61 (2000)
263. Mehta, N.R., Medvidovic, N.: Composing architectural styles from architectural primitives. In: ESEC / SIGSOFT FSE, pp. 347–350. ACM (2003). URL `http://dblp.uni-trier.de/db/conf/sigsoft/fse2003.html\#MehtaM03`
264. Menasce, D.A., Sousa, J.a.P., Malek, S., Gomaa, H.: Qos architectural patterns for self-architecting software systems. In: Proceedings of the 7th International Conference on Autonomic Computing, ICAC '10, pp. 195–204. ACM, New York, NY, USA (2010). DOI 10.1145/1809049.1809084
265. Metcalfe, J., Shimamura, A.P. (eds.): Metacognition: Knowing about knowing. MIT Press, Cambridge, MA, USA (1994)
266. Michalski, R.S.: A Theory and Methodology of Inductive Learning. In: Machine Learning, Symbolic Computation, pp. 83–134. Springer Berlin Heidelberg (1983)

267. Miettinen, K., Mäkelä, M.M.: Interactive bundle-based method for nondifferentiable multiobjeective optimization: nimbus §. Optimization Journal **34**(3), 231–246 (1995)
268. Minku, L.L.: Online ensemble learning in the presence of concept drift. Ph.D. thesis, School of Computer Science, University of Birmingham, Birmingham, UK (2010)
269. Minku, L.L., Yao, X.: DDD: A new ensemble approach for dealing with concept drift. IEEE Transactions on Knowledge and Data Engineering **24**(4), 619–633 (2012)
270. Minku, L.L., Yao, X.: Software effort estimation as a multi-objective learning problem. ACM Transactions on Software Engineering and Methodology **22**(4), 35:1–32 (2013)
271. Miranda, E.R., Wanderley, M.: New Digital Musical Instruments: Control And Interaction Beyond the Keyboard. A-R Editions, Inc., Middleton, WI (2006)
272. Mirollo, R.E., Strogatz, S.H.: Synchronization of pulse-coupled biological oscillators. SIAM Journal on Applied Mathematics **50**(6), 1645–1662 (1990)
273. Mitchell, M.: Self-awareness and control in decentralized systems. In: Metacognition in Computation, pp. 80–85 (2005)
274. Modler, P.: Neural networks for mapping hand gestures to sound synthesis parameters, vol. 18. IRCAM — Centre Pompidou (2000)
275. Moens, B., van Noorden, L., Leman, M.: D-jogger: Syncing music with walking. In: Sound and Music Computing Conference, pp. 451–456. Barcelona, Spain (2010)
276. Morin, A.: Levels of consciousness and self-awareness : A comparison and integration of various neurocognitive views. Cons. and Cog. **15**(2), 358–71 (2006)
277. Morin, A., Everett, J.: Conscience de soi et langage interieur: Quelques speculations. [self-awareness and inner speech: Some speculations]. Philosophiques **XVII**(2), 169–188 (1990)
278. Müller-Schloer, C., Schmeck, H., Ungerer, T.: Organic computing: a paradigm shift for complex systems. Springer (2011)
279. Nakashima, H., Aghajan, H., Augusto, J.C.: Handbook of ambient intelligence and smart environments. Springer (2009)
280. Narukawa, K., Tanigaki, Y., Ishibuchi, H.: Evolutionary many-objective optimization using preference on hyperplane. In: Proceedings of the 2014 Conference on Genetic and Evolutionary Computation Companion, pp. 91–92. ACM (2014)
281. Natarajan, P., Atrey, P.K., Kankanhalli, M.: Multi-camera coordination and control in surveillance systems: A survey. ACM Transactions on Multimedia Computing, Communications and Applications **11**(4), 57:1–57:30 (2015). DOI 10.1145/2710128
282. Nebehay, G., Chibamu, W., Lewis, P.R., Chandra, A., Pflugfelder, R., Yao, X.: Can diversity amongst learners improve online object tracking? In: Z.H. Zhou, F. Roli, J. Kittler (eds.) Multiple Classifier Systems, *Lecture Notes in Computer Science*, vol. 7872, pp. 212–223. Springer, Berlin / Heidelberg (2013). DOI 10.1007/978-3-642-38067-9_19
283. Nebehay, G., Pflugfelder, R.: Consensus-based matching and tracking of keypoints for object tracking. In: Proceedings of the Winter Conference on Applications of Computer Vision (WACV). IEEE (2014)
284. Nebro, A.J., Luna, F., Alba, E., Beham, A., Dorronsoro, B.: AbYSS: adapting scatter search for multiobjective optimization. Tech. Rep. ITI-2006-2, Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, Malaga (2006)
285. Neisser, U.: The roots of self-knowledge: Perceiving self, it, and thou. Annals of the NY AoS. **818**, 19–33 (1997)
286. netem. URL `http://www.linuxfoundation.org/collaborate/workgroups/networking/netem`. Website: `http://www.linuxfoundation.org/collaborate/workgroups/networking/netem`, (accessed January 2015)
287. Nguyen, A., Satish, N., Chhugani, J., Kim, C., Dubey, P.: 3.5-d blocking optimization for stencil computations on modern CPUs and GPUs. In: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–13. IEEE (2010)
288. Niezen, G., Hancke, G.P.: Evaluating and optimising accelerometer-based gesture recognition techniques for mobile devices. In: AFRICON, 2009. AFRICON'09., pp. 1–6. IEEE (2009)

289. Nishida, K.: Learning and detecting concept drift. Ph.D. thesis, Hokkaido University (2008). URL http://lis2.huie.hokudai.ac.jp/~knishida/paper/nishida2008-dissertation.pdf
290. Nishida, K., Yamauchi, K.: Detecting concept drift using statistical testing. In: Proceedings of the Tenth International Conference on Discovery Science (DS'07) - Lecture Notes in Artificial Intelligence, vol. 3316, pp. 264–269. Sendai, Japan (2007)
291. Niu, X., Chau, T.C.P., Jin, Q., Luk, W., Liu, Q.: Automating elimination of idle functions by run-time reconfiguration. In: 21st IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, FCCM 2013, Seattle, WA, USA, April 28-30, 2013, pp. 97–104 (2013)
292. Niu, X., Coutinho, J.G.F., Luk, W.: A scalable design approach for stencil computation on reconfigurable clusters. In: 23rd International Conference on Field programmable Logic and Applications, FPL 2013, Porto, Portugal, September 2-4, 2013, pp. 1–4 (2013)
293. Niu, X., Tsoi, K.H., Luk, W.: Reconfiguring distributed applications in FPGA accelerated cluster with wireless networking. In: International Conference on Field Programmable Logic and Applications, FPL 2011, September 5-7, Chania, Crete, Greece, pp. 545–550 (2011)
294. Niu, X., et al.: Exploiting run-time reconfiguration in stencil computation. In: Proc. FPL, pp. 173–180 (2012)
295. Niu, X., et al.: Exploiting run-time reconfiguration in stencil computation. In: Proc. FPL, pp. 173—-180 (2012)
296. NVIDIA: Cuda zone. http://www.nvidia.com/object/cuda_home_new.html. Online; accessed 2-April-2015
297. Nymoen, K., Chandra, A., Glette, K., Torresen, J.: Decentralized harmonic synchronization in mobile music systems. In: Proceedings of the International Conference on Awareness Science & Technology (iCAST), pp. 1–6 (2014)
298. Nymoen, K., Chandra, A., Glette, K., Torresen, J., Voldsund, A., Jensenius, A.R.: PheroMusic: Navigating a musical space for active music experiences. In: Proc. Int. Computer Music Conference (ICMC) joint with the Sound and Music Computing Conference, pp. 1715–1718 (2014)
299. Nymoen, K., Song, S., Hafting, Y., Torresen, J.: Funky Sole Music: Gait recognition and adaptive mapping. In: Proc. Int. Conf. on New Interfaces for Musical Expression (NIME), pp. 299–302 (2014)
300. Okuma, K., Taleghani, A., de Freitas, N., Little, J., Lowe, D.: A Boosted Particle Filter: Multitarget Detection and Tracking. In: Proceedings of 8th European Conference on Computer Vision, vol. 3021, pp. 28–39 (2004)
301. Olfati-Saber, R.: Distributed kalman filtering for sensor networks. In: Proceedings of the Conference on Decision and Control, pp. 5492–5498 (2007). DOI 10.1109/CDC.2007.4434303
302. Olsson, R.A., Keen, A.W.: Remote procedure call. The JR Programming Language: Concurrent Programming in an Extended Java pp. 91–105 (2004)
303. Ong, Y.S., Nair, P.B., Keane, A.J.: Evolutionary optimization of computationally expensive problems via surrogate modeling. AIAA Journal **41**(4), 689–696 (2003)
304. Ontañón, S., Plaza, E.: Multiagent Inductive Learning: An Argumentation-based Approach. In: J. Fürnkranz, T. Joachims (eds.) Proceedings of the 27th International Conference on Machine Learning (ICML), pp. 839–846. Omnipress, Haifa, Israel (2010)
305. Oxford: Oxford dictionaries – adapt. http://www.oxforddictionaries.com/definition/english/adapt (Accessed in December 2014)
306. Oza, N.C.: Online bagging and boosting. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, pp. 2340–2345 (2005)
307. Oza, N.C., Russell, S.: Experimental comparisons of online and batch versions of bagging and boosting. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 359–364. ACM (2001)
308. Özuysal, M., Calonder, M., Lepetit, V., Fua, P.: Fast keypoint recognition using random ferns. Pattern Analysis and Machine Intelligence **32**(3), 448–461 (2010). DOI 10.1109/tpami.2009.23

309. Page, I., Luk, W.: Compiling occam into field-programmable gate arrays. In: International Conference on Field programmable Logic and Applications (FPL) (1991)
310. Papakonstantinou, A., Liang, Y., Stratton, J.A., Gururaj, K., Chen, D., Hwu, W.W., Cong, J.: Multilevel granularity parallelism synthesis on fpgas. In: IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines, FCCM 2011, Salt Lake City, Utah, USA, 1-3 May 2011, pp. 178–185 (2011)
311. Parashar, M., Hariri, S.: Autonomic computing: an overview. In: Proceedings of the 2004 international conference on Unconventional Programming Paradigms, UPP'04, pp. 257–269. Springer-Verlag, Berlin (2005)
312. Parsons, S., Bahsoon, R., Lewis, P.R., Yao, X.: Towards a better understanding of self-awareness and self-expression within software systems. Tech. Rep. CSR-11-03, University of Birmingham, School of Computer Science, UK (2011)
313. Paul, C., Bass, L., Kazman, R.: Software Architecture in Practice. MA: Addison-Wesley (1998)
314. Paul, C., Kazman, R., Klein, M.: Evaluating Software Architectures: Methods and Case Studies. Addison-Wesley (2002)
315. Paulson, L.: DARPA creating self-aware computing. IEEE Computer **36**(3), 24 (2003). DOI 10.1109/MC.2003.1185213
316. Peleg, A., Wilkie, S., Weiser, U.C.: Intel MMX for multimedia pcs. Communications of the ACM **40**(1), 24–38 (1997)
317. Perkowitz, M., Philipose, M., Fishkin, K., Patterson, D.J.: Mining models of human activities from the web. In: Proceedings of the 13th international conference on World Wide Web, pp. 573–582 (2004)
318. Perrone, M., et al.: Reducing data movement costs: Scalable seismic imaging on blue gene. In: IPDPS, pp. 320–329 (2012)
319. Perrone, M.P., Cooper, L.N.: When networks disagree: Ensemble methods for hybrid neural networks. Neural Networks for Speech and Image Processing, Chapman-Hall, New York pp. 126–142 (1993)
320. Peskin, C.S.: Mathematical aspects of heart physiology. Courant Institute of Mathematical Sciences, New York University New York (1975)
321. Pflugfelder, R., Bischof, H.: People Tracking across Two Distant Self-calibrated Cameras. In: Proceedings of International Conference on Advanced Video and Signal based Surveillance. IEEE Computer Society Press (2006)
322. Pflugfelder, R., Bischof, H.: Tracking across Non-overlapping Views Via Geometry. In: Proceedings of the International Conference on Pattern Recognition (2008)
323. Phelps, S., Mcburney, P., Parsons, S.: Evolutionary mechanism design: A review. Autonomous Agents and Multi-Agent Systems **21**(2), 237–264 (2010)
324. Piciarelli, C., Esterle, L., Khan, A., Rinner, B., Foresti, G.: Dynamic reconfiguration in camera networks: a short survey. IEEE Transactions on Circuits and Systems for Video Technology **PP**(99), 1–1 (2015). DOI 10.1109/TCSVT.2015.2426575. To appear
325. Polikar, R.: Ensemble based systems in decision making. IEEE Circuits and Systems Magazine **6**(3), 21–45 (2006)
326. Polikar, R., Udpa, L., Udpa, S., Honavar, V.: Learn++: An incremental learning algorithm for supervised neural networks. IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews **31**(4), 497–508 (2001)
327. P.T.Eugster, P.A.Felber, R.Guerraoui, A.M.Kermarrec: The many faces of publish/subscribe. ACM Computing Surveys **35**, 114–131 (2003)
328. Puckette, M.: Pure Data (Pd) (software). URL `\url{http://puredata.info}`
329. Pylvänäinen, T.: Accelerometer based gesture recognition using continuous hmms. In: Pattern Recognition and Image Analysis, pp. 639–646. Springer (2005)
330. Quaritsch, M., Kreuzthaler, M., Rinner, B., Bischof, H., Strobl, B.: Autonomous Multicamera Tracking on Embedded Smart Cameras. EURASIP Journal on Embedded Systems Volume 2007 **2007**(1), 35–45 (2007)

331. Rajko, S., Qian, G., Ingalls, T., James, J.: Real-time gesture recognition with minimal training requirements and on-line learning. In: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pp. 1–8. IEEE (2007)
332. Ramamurthy, S., Bhatnagar, R.: Tracking recurrent concept drift in streaming data using ensemble classifiers. In: Proceedings of the Sixth International Conference on Machine Learning and Applications (ICMLA), pp. 404–409. Cincinnati, Ohio (2007)
333. Rammer, I., Szpuszta, M.: Advanced. NET Remoting. Springer (2005)
334. Ramos, C., Augusto, J.C., Shapiro, D.: Ambient intelligence - the next step for artificial intelligence. Intelligent Systems, IEEE **23**(2), 15–18 (2008). DOI 10.1109/MIS.2008.19
335. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. MIT Press (2006)
336. Reason [computer software]. URL `\url{https://www.propellerheads.se}`
337. ReconOS: A programming model and OS for reconfigurable hardware (2013)
338. Reisslein, M., Rinner, B., Roy-Chowdhury, A.: Smart camera networks. IEEE Computer **47**(5), 26–28 (2014)
339. Reyes, R., de Sande, F.: Automatic code generation for gpus in llc. The Journal of Supercomputing **58**(3), 349–356 (2011)
340. Richter, U., Mnif, M., Branke, J., Müller-Schloer, C., Schmeck, H.: Towards a generic observer/controller architecture for organic computing. In: C. Hochberger, R. Liskowsky (eds.) INFORMATIK 2006 – Informatik für Menschen, *LNI*, vol. P-93, pp. 112–119. Bonner Köllen Verlag (2006)
341. Rietmann, M., Messmer, P., Nissen-Meyer, T., Peter, D., Basini, P., Komatitsch, D., Schenk, O., Tromp, J., Boschi, L., Giardini, D.: Forward and adjoint simulations of seismic wave propagation on emerging large-scale GPU architectures. In: SC Conference on High Performance Computing Networking, Storage and Analysis, SC '12, Salt Lake City, UT, USA - November 11 - 15, 2012, pp. 38, Owner = Xinyu, Timestamp = 2015.05.07 (2012)
342. Rinner, B., Esterle, L., Simonjan, J., Nebehay, G., Pflugfelder, R., Fernandez, G., Lewis, P.R.: Self-Aware and Self-Expressive Camera Networks. Computer **48**(7), 33–40 (2015)
343. Rinner, B., Winkler, T., Schriebl, W., Quaritsch, M., Wolf, W.: The evolution from single to pervasive smart cameras. In: Proceedings of the Second ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), pp. 1–10. IEEE Computer Society Press (2008). DOI 10.1109/ICDSC.2008.4635674
344. Rinner, B., Wolf, W.: Introduction to Distributed Smart Cameras. Proceedings of the IEEE **96**(10), 1565–1575 (2008). DOI 10.1109/JPROC.2008.928742
345. RNA: Recursive Network Architecture. URL `www.isi.edu/rna`. Website: `www.isi.edu/rna`, (accessed January 2015)
346. Rochat, P.: Five levels of self-awareness as they unfold in early life. Consciousness and Cognition **12**, 717–731 (2003)
347. Russell, S.J., Norvig, P.: Artificial Intelligence - A Modern Approach, 3 edn. Pearson Education (2010)
348. Saaty, T.L.: The Analytical Hierarchical Process. McGraw-Hill (1980)
349. Sakellari, G.: The cognitive packet network: A survey. The Computer Journal **53** (2010)
350. SanMiguel, J.C., Shoop, K., Cavallaro, A., Micheloni, C., Foresti, G.L.: Self-Reconfigurable Smart Camera Networks. IEEE Computer **47**(5), 67–73 (2014)
351. Santambrogio, M., Hoffmann, H., Eastep, J., Agarwal, A.: Enabling technologies for self-aware adaptive systems. In: 2010 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), pp. 149–156. IEEE (2010)
352. Santner, J., Leistner, C., Saffari, A., Pock, T., Bischof, H.: PROST: Parallel robust online simple tracking. In: Computer Vision and Pattern Recognition, pp. 723–730 (2010)
353. Schaumeier, J., Jeremy Pitt, J., Cabri, G.: A tripartite analytic framework for characterising awareness and self-awareness in autonomic systems research. In: Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2012 Sixth IEEE Conference on, pp. 157–162 (2012)
354. Schlömer, T., Poppinga, B., Henze, N., Boll, S.: Gesture recognition with a wii controller. In: Proceedings of the 2nd international conference on Tangible and embedded interaction, pp. 11–14. ACM (2008)

355. Schmeck, H.: Organic computing - a new vision for distributed embedded systems. In: Object-Oriented Real-Time Distributed Computing (ISORC), Eighth IEEE International Symposium on, pp. 201–203. IEEE (2005)
356. Schmickl, T., Thenius, R., Moslinger, C., Timmis, J., Tyrrell, A., Read, M., Hilder, J., Halloy, J., Campo, A., Stefanini, C., et al.: Cocoro–the self-aware underwater swarm. In: Proc. Int. Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW), pp. 120–126. IEEE Computer Society, Ann Arbor, MI, USA (2011)
357. Schnier, T., Yao, X.: Using negative correlation to evolve fault-tolerant circuits. In: Proceedings of the 5th International Conference on Evolvable Systems: From Biology to Hardware (ICES'2003) – Lecture Notes in Computer Science, vol. 2606, pp. 35–46. Springer-Verlag (2003)
358. Scholz, M., Klinkenberg, R.: Boosting classifiers for drifting concepts. Intelligent Data Analysis **11**(1), 3–28 (2007)
359. Sharan, K.: Java remote method invocation. In: Beginning Java 8 APIs, Extensions and Libraries, pp. 525–548. Springer (2014)
360. Shaw, M.J., Sikora, R.: A distributed problem-solving approach to inductive learning. Tech. Rep. CMU-RI-TR-90-262, School of Computer Science, Carnegie Mellon University (1990)
361. Shipp, C.A., Kuncheva, L.I.: Relationships between combination methods and measures of diversity in combining classifiers. Information Fusion **3**(2), 135–148 (2002)
362. Showerman, M., et al.: QP: A heterogeneous multi-acceleator cluster. In: Proc. ICHPCC (2009)
363. Shukla, S.K., Yang, Y., Bhuyan, L.N., Brisk, P.: Shared memory heterogeneous computation on pcie-supported platforms. In: 23rd International Conference on Field programmable Logic and Applications, FPL 2013, Porto, Portugal, September 2-4, 2013, pp. 1–4 (2013)
364. Simonjan, J., Esterle, L., Rinner, B., Nebehay, G., Dominguez, G.F.: Demonstrating autonomous handover in heterogeneous multi-camera systems. In: Proceedings of the International Conference on Distributed Smart Cameras, pp. 43:1–43:3 (2014). DOI 10.1145/2659021.2669474
365. Sironi, F., Bartolini, D.B., Campanoni, S., Cancare, F., Hoffmann, H., Sciuto, D., Santambrogio, M.D.: Metronome: Operating System Level Performance Management via Self-adaptive Computing. In: Proc. Design Automation Conference (DAC). ACM (2012)
366. Sironi, F., Cuoccio, A., Hoffmann, H., Maggio, M., Santambrogio, M.: Evolvable Systems on Reconfigurable Architecture via Self-aware Adaptive Applications. In: NASA/ESA Conference on Adaptive Hardware and Systems (AHS) (2011). DOI 10.1109/AHS.2011.5963933
367. Sironi, F., Triverio, M., Hoffmann, H., Maggio, M., Santambrogio, M.: Self-aware Adaptation in FPGA-based Systems. In: Int. Conf. on Field Programmable Logic and Applications. IEEE (2010)
368. Smallwood, J., McSpadden, M., Schooler, J.: The lights are on but no one's home: meta-awareness and the decoupling of attention when the mind wanders. Psychonomic Bulletin and Review **14**(3), 527–533 (2007)
369. Song, S., Chandra, A., Torresen, J.: An ant learning algorithm for gesture recognition with one-instance training. In: Proc. Int. Congr. on Evolutionary Computation (CEC), pp. 2956–2963. IEEE (2013)
370. SRC Computers, L.: Src-7 mapstation. Tech. rep., SRC Computers, LLC (2009)
371. Srinivas, N., Deb, K.: Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation **2**(3), 221–248 (1994)
372. Stanley, K.O.: Learning concept drift with a committee of decision trees. Tech. Rep. UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin (2003)
373. Sterritt, R., Parashar, M., Tianfield, H., Unland, R.: A concise introduction to autonomic computing. Advanced Engineering Informatics **19**(3), 181–187 (2005)
374. Steuer, R.E., Choo, E.U.: An interactive weighted tchebycheff procedure for multiple objective programming. Mathematical Programming **26**(3), 326–344 (1983)
375. Stone, P.: Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer. MIT Press (2000)

376. Strassen, V.: Gaussian elimination is not optimal. Numerische Mathematik pp. 13:354–356 (1969)
377. Street, W., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: Proceedings of the Seventh ACM International Conference on Knowledge Discovery and Data Mining (KDD), pp. 377–382. ACM Press, New York (2001)
378. Strenski, D.: The cray xd1 computer and its reconfigurable architecture. Tech. rep., Cray Inc. (2005)
379. Strey, A., Bange, M.: Performance analysis of intel's MMX and SSE: A case study. In: Proceedings of 7th International Euro-Par Conference on Parallel Processing (Euro-Par), pp. 142–147. Manchester, UK (2001)
380. Susanto, K.W., Todman, T., Coutinho, J.G.F., Luk, W.: Design validation by symbolic simulation and equivalence checking: A case study in memory optimization for image manipulation. In: SOFSEM, *LNCS*, vol. 5404, p. 509–520. Springer (2009)
381. Sutter, H.: The free lunch is over: A fundamental turn toward concurrency in software. Dr. Dobb's Journal (2005)
382. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (1998)
383. Taj, M., Cavallaro, A.: Distributed and decentralized multi-camera tracking. Signal Processing Magazine **28**(3), 46–58 (2011)
384. Tawney, G.A.: Feeling and self-awareness. Psyc. Rev. **9**(6), 570 – 596 (1902)
385. Tesauro, G.: Reinforcement learning in autonomic computing: A manifesto and case studies. IEEE Internet Computing **11**(1), 22–30 (2007)
386. Thomas, D., Luk, W.: Non-uniform random number generation through piecewise linear approximations. In: Proc. FPL, pp. 1—-6 (2006)
387. Thomas, D.B., Luk, W.: Credit risk modelling using hardware accelerated monte-carlo simulation. In: 16th IEEE International Symposium on Field-Programmable Custom Computing Machines, FCCM 2008, 14-15 April 2008, Stanford, Palo Alto, California, USA, pp. 229–238 (2008)
388. Todman, T., Boehm, P., Luk, W.: Verification of streaming hardware and software codesigns. In: Proc. Int. Conf. on Field Programmable Technology (ICFPT), pp. 147–150. IEEE (2012)
389. Todman, T., Stilkerich, S.C., Luk, W.: Using statistical assertions to guide self-adaptive systems. International Journal of Reconfigurable Computing **2014**, 724,585.1–8 (2014). DOI 10.1155/2014/724585
390. Tong, X., Ngai, E.: A ubiquitous publish/subscribe platform for wireless sensor networks with mobile mules. In: Proceedings of the IEEE Eighth International Conference on Distributed Computing in Sensor Systems (DCOSS), 2012, pp. 99–108. IEEE (2012)
391. Torresen, J., Hafting, Y., Nymoen, K.: A new Wi-Fi based platform for wireless sensor data collection. In: Proceedings of the International Conference on New Interfaces for Musical Expression, pp. 337–340 (2013)
392. Touch, J., Pingali, V.: The RNA Metaprotocol. In: Proceedings of the International Conference on Computer Communications and Networks, pp. 1–6 (2008). DOI 10.1109/ICCCN.2008.ECP.46
393. Trucco, E., Plakas, K.: Video Tracking: A Concise Survey. Journal of Oceanic Engineering **31**(2), 520–529 (2006)
394. Tse, A.H.T., Chow, G.C.T., Jin, Q., Thomas, D.B., Luk, W.: Optimising performance of quadrature methods with reduced precision. In: Proc. Int. Conf. on Reconfigurable Computing: Architectures, Tools and Applications (ARC), *Lecture Notes in Computer Science*, vol. 7199, pp. 251–263. Springer, Berlin / Heidelberg (2012). DOI 10.1007/978-3-642-28365-9_21
395. Tse, A.H.T., Thomas, D.B., Tsoi, K.H., Luk, W.: Dynamic scheduling monte-carlo framework for multi-accelerator heterogeneous clusters. In: Proceedings of the International Conference on Field-Programmable Technology, FPT 2010, 8-10 December 2010, Tsinghua University, Beijing, China, pp. 233–240 (2010)
396. Tsoi, K.H., Luk, W.: Axel: A heterogeneous cluster with fpgas and gpus. In: Proc. FPGA, pp. 115–124 (2010)

397. Tsymbal, A., Pechenizkiy, M., Cunningham, P., Puuronen, S.: Dynamic integration of classifiers for handling concept drift. Information Fusion **9**(1), 56–68 (2008)
398. Vassev, E., Hinchey, M.: Knowledge representation and awareness in autonomic service-component ensembles – state of the art. In: 14th IEEE International Symposium on Object/Component/Service-oriented Real-time Distributed Computing, pp. 110–119. IEEE Computer Society (2011)
399. Vasudevan, S.: What is assertion-based verification? SIGDA E-News **42**(12) (2012)
400. Vermorel, J., Mohri, M.: Multi-Armed Bandit Algorithms and Empirical Evaluation. In: Proceedings of the European Conference on Machine Learning, pp. 437–448. Springer (2005)
401. Vickrey, W.: Counterspeculation, auctions, and competitive sealed tenders. The Journal of Finance **16**(1), 8–37 (1961)
402. Vinoski, S.: Corba: Integrating diverse applications within distributed heterogeneous environments. IEEE Communications Magazine **35**(2), 46–55 (1997)
403. Volker, L., Martin, D., El Khayaut, I., Werle, C., Zitterbart, M.: A Node Architecture for 1000 Future Networks. In: IEEE International Conference onCommunications (ICC), pp. 1–5 (2009). DOI 10.1109/ICCW.2009.5207996
404. Volker, L., Martin, D., Werle, C., Zitterbart, M., El-Khayat, I.: Selecting Concurrent Network Architectures at Runtime. In: IEEE International Conference on Communications (ICC), pp. 1–5 (2009). DOI 10.1109/ICC.2009.5199445
405. Wang, J., Brady, D., Baclawski, K., Kokar, M., Lechowicz, L.: The use of ontologies for the self-awareness of the communication nodes. In: Proceedings of the Software Defined Radio Technical Conference SDR, vol. 3 (2003)
406. Wang, S., Minku, L.L., Yao, X.: A learning framework for online class imbalance learning. In: Proceedings of the IEEE Symposium on Computational Intelligence and Ensemble Learning (CIEL), pp. 36–45 (2013)
407. Wang, S., Minku, L.L., Yao, X.: Online class imbalance learning and its applications in fault detection. International Journal of Computational Intelligence and Applications **12**(4), 1340,001(1–19) (2013)
408. Wang, S., Minku, L.L., Yao, X.: A multi-objective ensemble method for online class imbalance learning. In: Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN), pp. 3311–3318. IEEE (2014). DOI 10.1109/IJCNN.2014.6889545
409. Wang, S., Minku, L.L., Yao, X.: Resampling-based ensemble methods for online class imbalance learning. In: IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 27, pp. 1356–1368. IEEE (2015). DOI 10.1109/TKDE.2014.2345380
410. Wang, Z., Tang, K., Yao, X.: A memetic algorithm for multi-level redundancy allocation. IEEE Transactions on Reliability **59**(4), 754–765 (2010)
411. Watson, R.: The Delta-t Transport Protocol: Features and Experience. In: Local Computer Networks, 1989., Proceedings 14th Conference on, pp. 399–407 (1989). DOI 10.1109/LCN.1989.65288
412. Werner-Allen, G., Tewari, G., Patel, A., Welsh, M., Nagpal, R.: Firefly-inspired sensor network synchronicity with realistic radio effects. In: Proceedings of the 3rd international conference on Embedded networked sensor systems, pp. 142–153 (2005)
413. Weyns, D., Schmerl, B., Grassi, V., Malek, S., Mirandola, R., Prehofer, C., Wuttke, J., Andersson, J., Giese, H., Gäschka, K.M.: On patterns for decentralized control in self-adaptive systems. In: R. Lemos, H. Giese, H. Müller, M. Shaw (eds.) Software Engineering for Self-Adaptive Systems II, *Lecture Notes in Computer Science*, vol. 7475, pp. 76–107. Springer Berlin Heidelberg (2013)
414. Wikipedia: Wikipedia– adaptation (computer science). `http://en.wikipedia.org/wiki/Adaptation(CHANGEURL)` (Accessed in December 2014)
415. Winfield, A.: Robots with internal models: a route to self-aware and hence safer robots. In: J. Pitt (ed.) The Computer After Me. Imperial College Press / World Scientific Book (2014)
416. Wirsing, M., Hölzl, M., Koch, N., Mayer, P.: Software Engineering for Collective Autonomic Systems: The ASCENS Approach, *Lecture Notes in Computer Science*, vol. 8998. Springer (2015)

417. Wolf, W., Ozer, B., Lv, T.: Smart Cameras as Embedded Systems. IEEE Computer **35**(9), 48–53 (2002)
418. Wright, M.: Open Sound Control: an enabling technology for musical networking. Organised Sound **10**(3), 193–200 (2005)
419. Xiao, L., Zhu, Y., Ni, L., Xu, Z.: Gridis: An incentive-based grid scheduling. In: Proceedings of the 19th IEEE InternationalParallel and Distributed Processing Symposium, p. 65b (2005). DOI 10.1109/IPDPS.2005.237
420. Xilinx: Sdaccel development environment. `http://www.xilinx.com/products/design-tools/sdx/sdaccel.html`. [Online; accessed 2-April-2015]
421. Y. Jin et al: A framework for evolutionary optimization with approximate fitness functions. IEEE Transactions on Evolutionary Computation **6**(5), 481—-494 (2002)
422. Ye, J., Dobson, S., McKeever, S.: Situation identification techniques in pervasive computing: A review. Pervasive and Mobile Computing **8**(1) (2012)
423. Yiannacouras, P., Steffan, J.G., Rose, J.: VESPA: portable, scalable, and flexible fpga-based vector processors. In: Proceedings of the International Conference on Compilers, Architecture, and Synthesis for Embedded Systems, pp. 61–70 (2008)
424. Yilmaz, A., Javed, O., Shah, M.: Object Tracking: A Survey. Computing Surveys **38**(4) (2006)
425. Yin, F., D., M., Velastin, S.: Performance evaluation of object tracking algorithms. In: Proceedings of the International Workshop on Performance Evaluation of Tracking and Surveillance (2007)
426. Yin, L., Dong, M., Duan, Y., Deng, W., Zhao, K., Guo, J.: A high-performance training-free approach for hand gesture recognition with accelerometer. Multimedia Tools and Applications pp. 1–22 (2013)
427. Yu, X., Tang, K., Chen, T., Yao, X.: Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization. Memetic Computing **1**(1), 3–24 (2009)
428. Zadeh, L.: Optimality and non-scalar-valued performance criteria. IEEE Transactions on Automatic Control **8**(1), 59–60 (1963)
429. Zagal, J.C., Lipson, H.: Towards self-reflecting machines: Two-minds in one robot. In: Advances in Artificial Life. Darwin Meets von Neumann, *Lecture Notes in Computer Science*, vol. 5777, pp. 156–164. Springer Berlin Heidelberg (2011)
430. Zambonelli, F., Bicocchi, N., Cabri, G., Leonardi, L., Puviani, M.: On self-adaptation, self-expression, and self-awareness in autonomic service component ensembles. In: Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2011 Fifth IEEE Conference on, pp. 108 –113 (2011)
431. Zarezadeh, A.A., Bobda, C.: Hardware Middleware for Person Tracking on Embedded Distributed Smart Cameras. Hindawi International Journal of Reconfigurable Computing (2012)
432. Zeppenfeld, J., Bouajila, A., Stechele, W., Bernauer, A., Bringmann, O., Rosenstiel, W., Herkersdorf, A.: Applying ASoC to Multi-core Applications for Workload Management. In: C. Müller-Schloer, H. Schmeck, T. Ungerer (eds.) Organic Computing — A Paradigm Shift for Complex Systems, *Autonomic Systems*, vol. 1, pp. 461–472. Springer Basel (2011)
433. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation **1**(1), 32–49 (2011)
434. Ziliani, F., Velastin, S., Porikli, F., Marcenaro, L., Kelliher, T., Cavallaro, A., Bruneaut, P.: Performance evaluation of event detection solutions: the CREDS experience. In: Proceedings of the International Conference on Advanced Video and Signal Based Surveillance, pp. 201–206 (2005)
435. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithm: Empirical results. Evolutionary Computation **8**(2), 173–195 (2000)
436. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. Tech. Rep. 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich (2001)

437. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Transactions on Evolutionary Computation **3**(4), 257–271 (1999)
438. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on Evolutionary Computation **7**(2), 117–132 (2003)