On Scalability Issues in Reinforcement Learning for Self-Reconfiguring Modular Robots

Paulina Varshavskaya, Leslie Pack Kaelbling and Daniela Rus
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA
Email: paulina,lpk,rus@csail.mit.edu

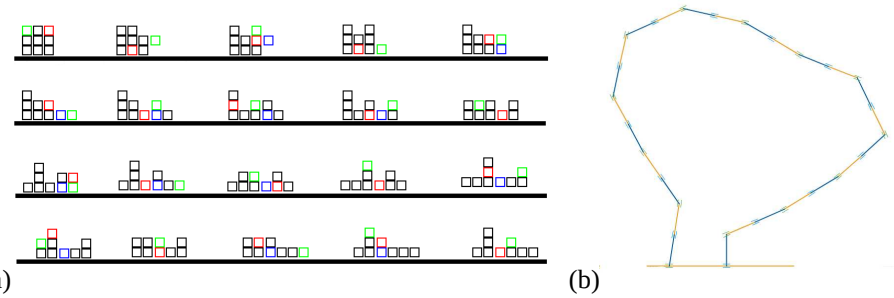
Self-reconfiguring modular robots have been receiving great attention because advances in our field are expected to deliver ultra-adaptable and robust systems. There has been remarkable progress in modular hardware and distributed controllers, e.g., [1]–[4], some of which were designed automatically by genetic algorithms, e.g., [1]. But how can the greatest adaptability be achieved? Our position is that modular robots need to run learning algorithms in order to adapt to the changing environment and deliver on the self-organization promise without (much) interference from human designers, programmers and operators.

We have developed a reinforcement learning (RL) approach to learning in self-reconfiguring modular robots. There are many scalability challenges in applying RL to our filed. A large number of modules means a large number of learning agents which modify their behavior at the same time, making the underlying process nonstationary. Local policies executed by individual modules need to give rise to coherent global behavior; as the number of modules increases, this property is hard to achieve both by human designers and learning algorithms. Finally, there is a tremendous growth of search spaces as a function of the number of modules in operation. We have been researching techniques to address these scalability issues. Specifically, we have developed two ways to dramatically reduce search spaces and thus simplify the learning problem: an incremental approach to learning, which is made possible specifically by the intrinsic modularity of our systems, and a log-linear representation which can be more universally used. Our results suggest that the learning algorithms could become scalable and produce large, adaptive systems.

I. Learning by gradient ascent in policy space

Learning even a simple task in the self-reconfigurable setting is formally hard whenever each module can only observe the environment locally and cannot know the global state of the system. Such situations are formally described by POMDPs and can be optimized by direct policy search. We take this approach with an algorithm called Gradient Ascent in Policy Space (GAPS), developed by [5]. In applying the GAPS algorithm to self-reconfiguring modular robots, we make the following assumptions about each robotic module: a local observation field, an ability to execute a number of actions, and sufficient computational power and memory to run the algorithm effectively. We have applied this learning algorithm to the problem of locomotion by self-reconfiguration in lattice-based robots in [6].

Policies are represented as lookup tables of parameters θ, one for each possible observation and action. The probability of taking a certain action given the current observation is then given by the standard Boltzmann's law. The algorithm is derived from the gradient of the value of a policy π_θ, which is $V_\theta = E_\theta[R(h)] = \sum_{h \epsilon H} R(h) P(h|\theta)$, where θ is the parameter vector defining the policy and H is the set of all possible experience histories. We do not have a model of the world that would give us $P(h|\theta)$ and so we use stochastic gradient ascent to locally maximize the value with the updates: $\frac{\partial}{\partial \theta_k} V_\theta = \sum_{h \epsilon H} R(h) \left(P(h|\theta) \sum_{t=1}^{T} \frac{\partial}{\partial \theta_k} \ln \pi_\theta(a_t, o_t) \right)$ for a learning episode of T timesteps (see [5] for a complete derivation). In our experiments, even in a two-dimensional simulator, which admitted eight local observation sites and nine possible actions,



(a) (b)

Fig. 1. (a) A 2D simulator of the robotic Molecule [4] moving East. (b) MultiShady [7] forming a closed chain.

the policy search space had thousands of parameters, which made any learning extremely lengthy. Our research focus has thus been on structuring the learning spaces specifically for self-reconfiguring modular robots and therefore making RL scalable for robots made out of many modules.

II. Structuring the search space

First, it is possible to leverage off the modular nature of our problem in order to make RL faster and more scalable. For the locomotion by self-reconfiguration task, we notice that the learning problem is easier when the robot has less modules than the size of its neighborhood, since it means that each module will see and have to learn a policy for a smaller number of states. If we start with only two modules, and add more incrementally, we effectively reduce the problem state-space. The problem becomes more manageable. Therefore, we have proposed the Incremental GAPS (IGAPS) algorithm [6], which takes as input the policy parameters to which a previous running instance of IGAPS with fewer modules has converged.

Second, instead of parameterizing the policy space by a huge lookup table where one parameter represents a single observation-action pair: $\theta_k = \theta_{o,a}$, we can represent the policy compactly as a function of a number of features defined over the observation-action space of the learning module. It is often the case that the designer can identify some salient parts of the observation that are important to the task being learned. This can be much easier than enumerating precisely which actions to take in which situations. We can define a vector of feature functions over the observation-action space $\Phi(a, o) = [\phi_1(a, o)\phi_2(a, o)...\phi_n(a, o)]$. These feature functions are domain-dependent and can have a discrete or continuous response field. Then the policy encoding for the learning agent is a modified version of Boltzmann's law, also known as a log-linear model. The resulting algorithm Log-linear GAPS (LLGAPS) is derived from the policy gradient value using the new represenation.

Our experimental results demonstrate that both approaches significantly reduce convergence times for the learning modules, while preserving the quality of learned policies. We have run experiments in locomotion by self-reconfiguration on a simulated lattice-based robot. Figure 1a shows a sequence executed by 9 modules after learning a policy for eastward locomotion with LLGAPS and 144 features. Figure 2a shows that learned policies achieve good rewards. We can also see there and in figure 2b that both incremental GAPS and a feature-based representation make the learning dramatically faster.

We are currently investigating other approaches to structuring the policy space for faster RL, as well as testing the developed techniques on a very different platform: a bipartite modular system of muntin-climbing robots called MultiShady [7]. Figure 1b shows the robot moving into a closed-chain configuration.

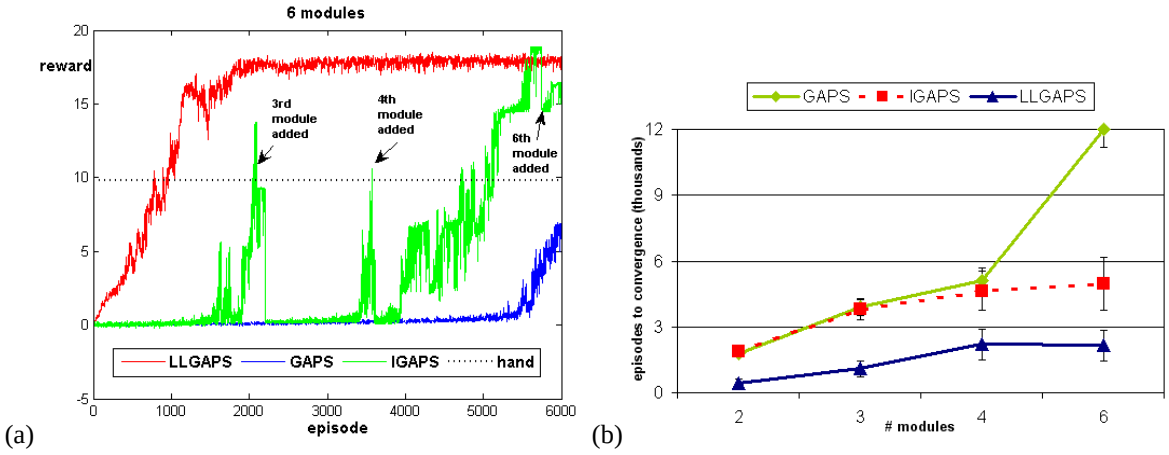


Fig. 2. Comparison between original GAPS, IGAPS, and LLGAPS. (a) Smoothed average rewards over 10 runs with 6 modules. (b) Mean convergence times for different numbers of modules.

References

[1] A. Kamimura, H. Kurokawa, E. Yoshida, S. Murata, K. Tomita, and S. Kokaji. Distributed adaptive locomotion by a modular robotic system, m-tran ii – from local adaptation to global coordinated motion using cpg controllers. In *Proc. of Int. Conference on Robots and Systems (IROS)*, 2004.

[2] V. Zykov, E. Mytilinaios, B. Adams, and H. Lipson. Self-reproducing machines. *Nature*, 435(7038):163–164, 2005.

[3] J. Bishop, S. Burden, E. Klavins, R. Kreisberg, W. Malone, N. Napp, and T. Nguyen. Self-organizing programmable parts. In *Proc. of the Int.Conf. on Robots and Systems (IROS)*, 2005.

[4] K. Kotay and D. Rus. Efficient locomotion for a self-reconfiguring robot. In *Proc. of Int. Conference on Robotics and Automation (ICRA)*, 2005.

[5] L. Peshkin. *Reinforcement Learning by Policy Search.* PhD thesis, Brown University, November 2001.

[6] P. Varshavskaya, L. P. Kaelbling, and D. Rus. Distributed learning for modular robots. In *Proc. Int. Conference on Robots and Systems (IROS)*, 2004.

[7] C. Detweiler, M. Vona, K. Kotay, and D. Rus. Hierarchical control for self-assembling mobile trusses with passive and active links. In *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*, 2006.