Twenty-Eighth International Conference on Automated Planning and Scheduling (ICAPS 2018)

Online Algorithms for POMDPs with Continuous State, Action, and Observation Spaces

Zachary N. Sunberg,* **Mykel J. Kochenderfer***
Aeronautics and Astronautics
Stanford University
Stanford, CA 94305

Abstract

Online solvers for partially observable Markov decision processes have been applied to problems with large discrete state spaces, but continuous state, action, and observation spaces remain a challenge. This paper begins by investigating double progressive widening (DPW) as a solution to this challenge. However, we prove that this modification alone is not sufficient because the belief representations in the search tree collapse to a single particle causing the algorithm to converge to a policy that is suboptimal regardless of the computation time. This paper proposes and evaluates two new algorithms, POMCPOW and PFT-DPW, that overcome this deficiency by using weighted particle filtering. Simulation results show that these modifications allow the algorithms to be successful where previous approaches fail.

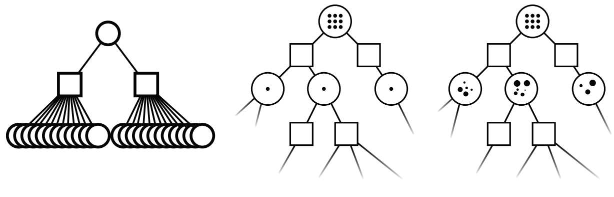

Figure 1: Tree Structure Comparison. Each square is an action node, and each circle is an observation node. Each black dot corresponds to a state particle with the size representing its weight. In continuous observation spaces, POMCP will create a tree that is too wide and shallow to be useful, and the beliefs in a POMCP-DPW tree will degenerate to a single particle, while POMCPOW maintains weighted particle mixture beliefs.

1 Introduction

The partially observable Markov decision process (POMDP) is a flexible mathematical framework for representing sequential decision problems (Littman, Cassandra, and Kaelbling 1995; Thrun, Burgard, and Fox 2005). Once a problem has been formalized as a POMDP, a wide range of solution techniques can be used to solve it. In a POMDP, at each step in time, an agent selects an action that causes the state to change stochastically to a new value. The agent seeks to maximize the expectation of the reward, which is a function of the state and action. However, the agent cannot directly observe the state, and makes decisions based only on observations that are stochastically generated by the state.

Solving large POMDPs generally requires the use of *online* methods (Silver and Veness 2010; Somani et al. 2013; Kurniawati and Yadav 2016). One widely used online algorithm is partially observable Monte Carlo planning (POMCP) (Silver and Veness 2010), which is an extension to Monte Carlo tree search that implicitly uses an unweighted particle filter to represent beliefs in the search tree.

POMCP and other online methods can accomodate continuous state spaces, and there has been recent work on solving problems with continuous action spaces (Seiler, Kurniawati, and Singh 2015). However, there has been less progress on problems with continuous observation spaces. This paper presents two similar algorithms which address the challenge of solving POMDPs with continuous state, action, and observation spaces. The first is based on POMCP and is called partially observable Monte Carlo planning with observation widening (POMCPOW). The second solves the belief-space MDP and is called particle filter trees with double progressive widening (PFT-DPW).

Two challenges make tree search difficult in continuous spaces. First, since the probability of sampling the same real number twice from a continuous random variable is zero, the width of the planning trees explodes on the first step, causing them to be too shallow to be useful (see Fig. 1). POMCPOW and PFT-DPW resolve this issue with a technique called double progressive widening (DPW) (Couëtoux et al. 2011). The second issue is that, even when DPW is applied, the belief representations used by current solvers collapse to a single state particle, resulting in overconfidence. As a consequence, the solutions obtained resemble QMDP policies, and there is no incentive for information gathering. POMCPOW and PFT-DPW overcome this issue by using the observation model to weight the particles used to represent beliefs.

This paper proceeds as follows: Section 2 contains background references, Section 3 presents the new algorithms, and Section 4 gives experimental results. More detail is con-

*{zsunberg, mykel}@stanford.edu

tained in the extended version of the paper (Sunberg and Kochenderfer 2018).

2 Background

Due to space restrictions, most background is left to the extended paper (Sunberg and Kochenderfer 2018). This text assumes familiarity with POMDPs, with $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{O}$ denoting the state, action, and observation spaces, $\mathcal{T}$, $\mathcal{Z}$, and $\mathcal{R}$ the transition, observation, and reward models, γ the discount factor, and G a generative model. It also assumes prior knowledge of Monte Carlo tree search (MCTS) and upper confidence trees (UCT) (Browne et al. 2012), DPW (Couëtoux et al. 2011), POMCP (Silver and Veness 2010), and particle filtering (Thrun, Burgard, and Fox 2005).

A variety of offline and online techniques are available for solving POMDPs (Kaelbling, Littman, and Cassandra 1998; Thrun 1999; Kurniawati, Hsu, and Lee 2008; Bai, Hsu, and Lee 2014; Brechtel, Gindele, and Dillmann 2013; Platt et al. 2010; Van Den Berg, Patil, and Alterovitz 2012; Agha-Mohammadi, Chakravorty, and Amato 2011; Melchior and Simmons 2007; Prentice and Roy 2009; Bry and Roy 2011; Ross et al. 2008; Silver and Veness 2010; Somani et al. 2013; Kurniawati and Yadav 2016; Pas 2012). However, there remains a need for simple, general purpose online POMDP solvers that can handle continuous spaces, especially continuous observation spaces.

3 Algorithms

This section presents several MCTS algorithms for continuous POMDPs including the new POMCPOW and PFT-DPW approaches. The three algorithms in this section share a common outer structure shown in the extended version (Sunberg and Kochenderfer 2018). The difference is in the SIMULATE procedure.

The following variables are used in the listings and text: h represents a history $(b, a_1, o_1, \ldots a_k, o_k)$, and ha and hao are shorthand for histories with a and (a, o) appended to the end, respectively; d is the depth to explore; Q is a value function estimate; C is a list of the children of a node (along with the reward in the case of PFT-DPW); N is a count of the number of visits; and M is the number of times that a history has been generated. The list of states associated with a node is denoted B, and W is a list of weights corresponding to those states.

3.1 POMCP-DPW

The first algorithm is POMCP with double progressive widening (POMCP-DPW), listed in Algorithm 1. DPW prevents excessive widening by rejecting some simulated observations. In the case of a rejection (line 12), the tree search is continued with a previously simulated observation (line 13) and a state is sampled from the associated belief (line 14).

This algorithm obtained remarkably good solutions for a very large autonomous freeway driving POMDP with multiple vehicles (Sunberg, Ho, and Kochenderfer 2017). To our knowledge, that is the first work applying progressive widening to POMCP.

Unfortunately, on continuous observation spaces, POMCP-DPW is suboptimal. In particular, the planner maximizes the QMDP value (Definition 1). This is expressed formally for a modified version of POMCP-DPW in Theorem 1 below.

Algorithm 1 POMCP-DPW

```
1:  procedure SIMULATE($s$, $h$, $d$)
2:      if $d = 0$
3:          return 0
4:      $a \leftarrow$ ACTIONPROGWIDEN($h$)
5:      if $|C(ha)| \leq k_o N(ha)^{\alpha_o}$
6:          $s', o, r \leftarrow G(s, a)$
7:          $C(ha) \leftarrow C(ha) \cup \{o\}$
8:          $M(hao) \leftarrow M(hao) + 1$
9:          append $s'$ to $B(hao)$
10:         if $M(hao) = 1$
11:             return $r + \gamma$ROLLOUT$(s', hao, d)$
12:     else
13:         $o \leftarrow$ select $o \in C(ha)$ w.p. $\frac{M(hao)}{\sum_o M(hao)}$
14:         $s' \leftarrow$ select $s' \in B(hao)$ w.p. $\frac{1}{|B(hao)|}$
15:         $r \leftarrow R(s, a, s')$
16:     $total \leftarrow r + \gamma$SIMULATE$(s', hao, d-1)$
17:     $N(h) \leftarrow N(h) + 1$
18:     $N(ha) \leftarrow N(ha) + 1$
19:     $Q(ha) \leftarrow Q(ha) + \frac{total - Q(ha)}{N(ha)}$
20:     return $total$
```

Definition 1 (QMDP value). *Let $Q_{MDP}(s, a)$ be the optimal state-action value function assuming full observability starting by taking action a in state s. The* QMDP value *at belief b, $Q_{MDP}(b, a)$, is the expected value of $Q_{MDP}(s, a)$ when s is distributed according to b.*

Theorem 1 (Modified POMCP-DPW convergence to QMDP). *If a bounded-horizon POMDP meets the following conditions: 1) the state and observation spaces are continuous with a finite observation probability density function, and 2) the regularity hypothesis is met, then modified POMCP-DPW will produce a value function estimate, $\hat{Q}$, that converges to the QMDP value for the problem. Specifically, there exists a constant $C > 0$, such that after n iterations,*

$$\left|\hat{Q}(b, a) - Q_{MDP}(b, a)\right| \leq \frac{C}{n^{1/(10 d_{\max} - 7)}}$$

exponentially surely in n, for every action a.

The extended version of this paper (Sunberg and Kochenderfer 2018) contains a proof of this theorem that leverages work by Auger, Couetoux, and Teytaud (2013). It also provides a complete description of the modified algorithm. The key idea is that belief nodes will contain only a single state particle (see Fig. 1) because the generative model will (with probability one) produce a unique observation o each time it is queried. Thus, for every generated history h, only one state will ever be inserted into $B(h)$ (line 9, Algorithm 1), and therefore h is merely an alias for that state. Since each belief node corresponds to a state, the solver is actually solving the fully observable MDP except at the root node, leading to a QMDP solution.

As a result of Theorem 1, modified POMCP-DPW will plan a QMDP policy (see Corollary 1 of Auger, Couetoux,

Algorithm 2 POMCPOW

```
1: procedure SIMULATE($s$, $h$, $d$)
2:     if $d = 0$
3:         return 0
4:     $a \leftarrow$ ACTIONPROGWIDEN($h$)
5:     $s', o, r \leftarrow G(s, a)$
6:     if $|C(ha)| \leq k_o N(ha)^{\alpha_o}$
7:         $M(hao) \leftarrow M(hao) + 1$
8:     else
9:         $o \leftarrow$ select $o \in C(ha)$ w.p. $\frac{M(hao)}{\sum_o M(hao)}$
10:    append $s'$ to $B(hao)$
11:    append $\mathcal{Z}(o \mid s, a, s')$ to $W(hao)$
12:    if $o \notin C(ha)$                              ▷ new node
13:        $C(ha) \leftarrow C(ha) \cup \{o\}$
14:        $total \leftarrow r + \gamma$ROLLOUT($s', hao, d-1$)
15:    else
16:        $s' \leftarrow$ select $B(hao)[i]$ w.p. $\frac{W(hao)[i]}{\sum_{j=1}^{m} W(hao)[j]}$
17:        $r \leftarrow R(s, a, s')$
18:        $total \leftarrow r + \gamma$SIMULATE($s', hao, d-1$)
19:    $N(h) \leftarrow N(h) + 1$
20:    $N(ha) \leftarrow N(ha) + 1$
21:    $Q(ha) \leftarrow Q(ha) + \frac{total - Q(ha)}{N(ha)}$
22:    return $total$
```

and Teytaud (2013)). For many problems this is a very useful solution, but since it neglects the value of information, it is suboptimal for problems where information gathering is important (Littman, Cassandra, and Kaelbling 1995; Kochenderfer 2015). Modified POMCP-DPW, POMCP-DPW, DESPOT, and ABT all share the characteristic that a belief node can only contain multiple states if they generated exactly the same observation. The experiments in Section 4 show that POMCP-DPW and DESPOT plan with similar performance to QMDP, and this is presumably the case for ABT as well.

3.2 POMCPOW

POMCPOW (Algorithm 2) is a new algorithm with weighted particle beliefs that expand gradually as more simulations are added. Since the number of particles in each belief node is related to the number of times the node is visited, nodes that are more likely to be reached by the optimal policy have richer belief representations. At each step, the simulated state is inserted into the weighted particle collection that represents the belief (line 10), and a new state is sampled from that belief (line 16). Figure 1 shows the tree structure contrast from POMCP-DPW. Because the resampling in line 16 can be efficiently implemented with binary search, the computational complexity is $\mathcal{O}(nd\log(n))$.

3.3 PFT-DPW

Another approach for solving continuous POMDPs online is MCTS-DPW on the equivalent belief MDP using particle filters to approximate the generative model. This new approach will be referred to as particle filter trees with double progressive widening (PFT-DPW). It is shown in Algorithm 3, where $G_{\text{PF}(m)}(b, a)$ is a particle filter belief update performed with a simulated observation and m state particles. The authors

Algorithm 3 PFT-DPW

```
1: procedure SIMULATE($b$, $d$)
2:     if $d = 0$
3:         return 0
4:     $a \leftarrow$ ACTIONPROGWIDEN($b$)
5:     if $|C(ba)| \leq k_o N(ba)^{\alpha_o}$
6:         $b', r \leftarrow G_{\text{PF}(m)}(b, a)$
7:         $C(ba) \leftarrow C(ba) \cup \{(b', r)\}$
8:         $total \leftarrow r + \gamma$ROLLOUT($b', d-1$)
9:     else
10:        $b', r \leftarrow$ sample uniformly from $C(ba)$
11:        $total \leftarrow r + \gamma$SIMULATE($b', d-1$)
12:    $N(b) \leftarrow N(b) + 1$
13:    $N(ba) \leftarrow N(ba) + 1$
14:    $Q(ba) \leftarrow Q(ba) + \frac{total - Q(ba)}{N(ba)}$
15:    return $total$
```

are not aware of any previous mention of this algorithm, but it is very likely that MCTS with particle filters has been used before without double progressive widening under another name.

PFT-DPW differs from POMCP and POMCPOW because it relies on simulating approximate belief trajectories instead of state trajectories. The primary shortcoming of this algorithm is that the number of particles in the filter, m, must be chosen a-priori and is static throughout the tree. Fortunately, the experiments in Section 4 show that it is often easy to choose m in practice.

3.4 Observation Distribution Requirement

It is important to note that, while POMCP, POMCP-DPW, and DESPOT only require a generative model of the problem, both POMCPOW and PFT-DPW also require $\mathcal{Z}$. We think that this is a reasonable requirement for two reasons. First, this requirement is no more stringent than the requirement for a standard importance resampling particle filter. Second, given the implications of Theorem 1, it is difficult to imagine a tree-based decision-making algorithm or a robust belief updater that does not require some way of measuring whether a state belongs to a belief or history. In practice, using POMCP or DESPOT to repeatedly observe and act in an environment already requires more than a generative model. For example, Silver and Veness (2010) use heuristic particle reinvigoration in lieu of $\mathcal{Z}$.

4 Experiments

Numerical simulation results are shown in Table 1. The Julia code (https://github.com/zsunberg/ContinuousPOMDPTreeSearchExperiments.jl) uses the POMDPs.jl framework (Egorov et al. 2017). Additional detail is provided the extended paper (Sunberg and Kochenderfer 2018).

4.1 Laser Tag

The Laser Tag benchmark is taken directly from the work of Somani et al. (2013) and included for the sake of calibration. DESPOT outperforms the other methods. The score for DESPOT differs slightly from that reported by Somani et al.

Table 1: Experimental Results

	Laser Tag (D, D, D)		Light Dark (D, D, C)		Sub Hunt (D, D, C)		VDP Tag (C, C, C)	
POMCPOW	-10.3 ± 0.2	🟩	56.1 ± 0.6	🟩	69.2 ± 1.3	🟩	29.3 ± 0.8	🟩
PFT-DPW	-11.1 ± 0.2	🟩	57.2 ± 0.5	🟩	77.4 ± 1.1	🟩	27.2 ± 0.8	🟩
QMDP	-10.5 ± 0.2	🟩	-6.4 ± 1.0	🟥	28.0 ± 1.3	🟫		
POMCP-DPW	-10.6 ± 0.2	🟩	-7.3 ± 1.0	🟥	28.3 ± 1.3	🟫	16.4 ± 1.0	🟫
DESPOT	-8.9 ± 0.2	🟩	-6.8 ± 1.0	🟥	26.8 ± 1.3	🟫		
POMCP^{D}	-14.1 ± 0.2	🟫	61.1 ± 0.4	🟩	28.0 ± 1.3	🟫	14.7 ± 0.9	🟫
DESPOT^{D}			54.2 ± 1.1	🟩	27.4 ± 1.3	🟫	14.3 ± 1.0	🟫

The three C or D characters after the solver indicate whether the state, action, and observation spaces are continuous or discrete, respectively. Solvers with a superscript D were run on a version of the problem with discretized action and observation spaces.

(2013) likely because of bounds implementation differences. POMCP performs much better than reported by Somani et al. (2013) because this implementation uses a state-based rollout policy.

4.2 Light Dark

In the Light Dark domain, the agent moves over integers trying to find 0 and take a specific action there, but it only receives accurate measurements in the "light" region around 10. QMDP and solvers predicted to behave like QMDP attempt to move directly to the origin, while POMCPOW and PFT-DPW perform better. Discretization allows POMCP and DESPOT to perform well in this one dimensional case but fails in subsequent problems with more observation dimensions.

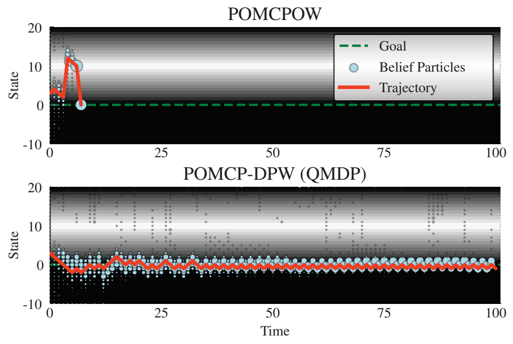

Figure 2: Example Light Dark trajectories.

4.3 Sub Hunt

In Sub Hunt, a submarine attempts to track and destroy an enemy sub. $\mathcal{S}$ and $\mathcal{A}$ are discrete so that QMDP can be used to solve the problem for comparison. The sub can choose to use active sonar to get better measurements at the cost of alerting the other submarine to its presence and reducing the probability of a successful engagement. The optimal strategy includes using the active sonar. Previous approaches have difficulty choosing to use active sonar because of the reduced engagement success rate and thus behave similarly to QMDP.

4.4 Van Der Pol Tag

Van Der Pol tag has continuous state, action, and observation spaces. In this problem an agent moves through 2D space with obstacles to try to tag a target that moves according to the Van Der Pol differential equations. This problem has several challenging features that might be faced in real-world applications. First, the state transitions are more computationally expensive because of the numerical integration. Second, the continuous state space and obstacles make it difficult to construct a good heuristic rollout policy, so random rollouts are used. POMCPOW and PFT-DPW outperform the other solvers in this case.

5 Conclusion

In this paper, we proposed new general-purpose online POMDP algorithms able to solve problems with continuous state, action, and observation spaces. This is a qualitative advance in capability over previous solution techniques. This study has yielded several insights. We explained why POMCP-DPW and other solvers are unable to choose costly information-gathering actions in continuous spaces, and showed that POMCPOW and PFT-DPW can overcome this challenge.

The theoretical properties of the algorithms remain to be proven. Additionally, better ways for choosing continuous actions (Seiler, Kurniawati, and Singh 2015; Mansley, Weinstein, and Littman 2011) would provide an improvement.

Acknowledgements Toyota Research Institute ("TRI") provided funds to assist the authors with their research, but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. The authors would also like to thank Zongzhang Zhang for his especially helpful comments.

References

Agha-Mohammadi, A.-A.; Chakravorty, S.; and Amato, N. M. 2011. FIRM: Feedback controller-based information-state roadmap - a framework for motion planning under uncertainty. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Auger, D.; Couetoux, A.; and Teytaud, O. 2013. Continuous upper confidence trees with polynomial exploration–consistency. In *Joint European Conference on Machine*

Learning and Knowledge Discovery in Databases, 194–209. Springer.

Bai, H.; Hsu, D.; and Lee, W. S. 2014. Integrated perception and planning in the continuous space: A POMDP approach. *International Journal of Robotics Research* 33(9):1288–1302.

Brechtel, S.; Gindele, T.; and Dillmann, R. 2013. Solving continuous POMDPs: Value iteration with incremental learning of an efficient space representation. In *International Conference on Machine Learning (ICML)*, 370–378.

Browne, C. B.; Powley, E.; Whitehouse, D.; Lucas, S. M.; Cowling, P. I.; Rohlfshagen, P.; Tavener, S.; Perez, D.; Samothrakis, S.; and Colton, S. 2012. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games* 4(1):1–43.

Bry, A., and Roy, N. 2011. Rapidly-exploring random belief trees for motion planning under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, 723–730.

Couëtoux, A.; Hoock, J.-B.; Sokolovska, N.; Teytaud, O.; and Bonnard, N. 2011. Continuous upper confidence trees. In *Learning and Intelligent Optimization*.

Egorov, M.; Sunberg, Z. N.; Balaban, E.; Wheeler, T. A.; Gupta, J. K.; and Kochenderfer, M. J. 2017. POMDPs.jl: A framework for sequential decision making under uncertainty. *Journal of Machine Learning Research* 18(26):1–5.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101:99–134.

Kochenderfer, M. J. 2015. *Decision Making Under Uncertainty: Theory and Application*. MIT Press.

Kurniawati, H., and Yadav, V. 2016. An online POMDP solver for uncertainty planning in dynamic environment. In *Robotics Research*. Springer. 611–629.

Kurniawati, H.; Hsu, D.; and Lee, W. S. 2008. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning (ICML)*.

Mansley, C. R.; Weinstein, A.; and Littman, M. L. 2011. Sample-based planning for continuous action Markov decision processes. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

Melchior, N. A., and Simmons, R. 2007. Particle RRT for path planning with uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*.

Pas, A. 2012. Simulation based planning for partially observable markov decision processes with continuous observation spaces. Master's thesis, Maastricht University.

Platt, Jr., R.; Tedrake, R.; Kaelbling, L.; and Lozano-Perez, T. 2010. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems*.

Prentice, S., and Roy, N. 2009. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research* 28(11-12):1448–1465.

Ross, S.; Pineau, J.; Paquet, S.; and Chaib-Draa, B. 2008. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research* 32:663–704.

Seiler, K. M.; Kurniawati, H.; and Singh, S. P. N. 2015. An online and approximate solver for POMDPs with continuous action space. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2290–2297.

Silver, D., and Veness, J. 2010. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems (NIPS)*.

Somani, A.; Ye, N.; Hsu, D.; and Lee, W. S. 2013. DESPOT: Online POMDP planning with regularization. In *Advances in Neural Information Processing Systems (NIPS)*, 1772–1780.

Sunberg, Z. N., and Kochenderfer, M. J. 2018. Online algorithms for POMDPs with continuous state, action, and observation spaces (extended version). https://arxiv.org/abs/1709.06196.

Sunberg, Z. N.; Ho, C. J.; and Kochenderfer, Mykel, J. 2017. The value of inferring the internal state of traffic participants for autonomous freeway driving. In *American Control Conference (ACC)*.

Thrun, S.; Burgard, W.; and Fox, D. 2005. *Probabilistic Robotics*. MIT Press.

Thrun, S. 1999. Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems (NIPS)*, volume 12, 1064–1070.

Van Den Berg, J.; Patil, S.; and Alterovitz, R. 2012. Motion planning under uncertainty using iterative local optimization in belief space. *International Journal of Robotics Research* 31(11):1263–1278.