ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αντιμετώπιση κακόβουλων απειλών εντός δικτύων μεγάλου μεγέθους, POMDPs και Ενισχυτική Μάθηση

Παναγιώτης Ν. Ράπτης

Επιβλέπων: **Νικόλαος Καλουπτσίδης,** Καθηγητής

ΑΘΗΝΑ

ΟΚΤΩΒΡΙΟΣ 2019

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αντιμετώπιση κακόβουλων απειλών εντός δικτύων μεγάλου μεγέθους, POMDPs και Ενισχυτική Μάθηση

Παναγιώτης Ν. Ράπτης

Α.Μ.: 1115 2014 00 170

ΕΠΙΒΛΕΠΩΝ: Νικόλαος Καλουπτσίδης, Καθηγητής

ΠΕΡΙΛΗΨΗ

Εδώ και αρκετά χρόνια, η παγκόσμια επιστημονική κοινότητα δίνει ιδιαιτερη σημασία σε μεθόδους μάθησης και λήψης αποφάσεων υπο αβεβαιότητα. Τα δεδομένα συνήθως είναι πολλά, η πολυπλοκότητα επίλυσης προβλημάτων πραγματικού κόσμου, υψηλή και η επιθυμία για την επίτευξη της καλύτερης δυνατής απόδοσης, δεδομένη. Σε όλο αυτό το πλαίσιο, προστίθενται και οι Μαρκοβιανές διαδικασίες αποφάσεων (MDPs, POMDPs), οι οποίες είναι ιδιαίτερα δημοφιλείς, εδώ και περίπου τρείς δεκαετίες. Ειδικότερα, οι μερικώς παρατηρούμενες Μαρκοβιανές διαδικασίες αποφάσεων (POMDPs), εκ φύσεως, δύνανται να μοντελοποιούν καλύτερα προβλήματα του πραγματικού κόσμου, όπως για παράδειγμα την επιτυχή περιήγηση ενός robot στον χώρο (*robot navigation*), αυτοματοποιημένα συστήματα οδήγησης, αναγνώριση φωνής, εύρεση στρατηγικής νίκης σε παιχνίδια όπως το σκάκι, το *Go* κ.α., έναντι των απλών Μαρκοβιανών διαδικασιών αποφάσεων (MDPs). Το βασικό μειονέκτημά τους, όμως, είναι το γεγονός ότι η βέλτιστη επίλυση ενός POMDP, στο πλαίσιο κάποιου προβλήματος του πραγματικού κόσμου, είναι στην πράξη υπολογιστικά ανέφικτη. Γίνεται, επομένως, σαφές ότι η υφιστάμενη υπολογιστική πολυπλοκότητα καθιστά ως μοναδική επιλογή την χρήση προσεγγιστικών αλγορίθμων. Εάν και υπαρχούν αρκετές διαφορετικές οικογένειες αλγορίθμων, οι οποίες επιλύουν προσεγγιστικά το POMDP πρόβλημα, σε αυτή την εργασία δίνεται ιδιαίτερη βαρύτητα σε μια συγκεκριμένη μέθοδο της οικογένειας των *policy-gradient* αλγορίθμων από την ενισχυτική μάθηση. Γενικά μιλώντας, η φιλοσοφία των *policy-gradient* μεθόδων στηρίζεται στην κατάλληλη προσαρμογή των παραμέτρων του υφιστάμενου *decision-maker (agent)*, έτσι ώστε να ελαχιστοποιείται το *long-term μέσο κόστος*, με το οποίο ζημιώνεται. Επίσης, οι *policy-gradient* μέθοδοι είναι ιδιαίτερα ελκυστικοί, καθώς αποτελούν μία υπολογιστικά εφικτή προσέγγιση της επίλυσης ενός POMDP μεγάλου μεγέθους.

Στο πλαίσιο της συγκεκριμένης πτυχιακής εργασίας, αρχικά γίνεται μια αναλυτική επισκόπηση ξεκινώντας από τα MDPs (Κεφάλαιο 1) και καταλήγοντας στην συνέχεια στα POMDPs (Κεφάλαιο 2), η βασική ιδέα των οποίων, αποτελεί επέκταση εκείνων που διέπουν τα MPDs. Πραγματοποιείται μια λεπτομερής, πλήν κουραστική, συζήτηση των βασικών εννοιών και μεθόδων που είναι άρρηκτα συνδεδεμένες με εκείνα, ενώ στο τέλος του δεύτερου κεφαλαίου παρουσιάζεται και η *"Αγία Τριάδα"* αλγορίθμων, οι οποίοι, κατά την γνώμη μας, δύνανται να επιλύουν προβλήματα POMDPs αρκετά μεγάλου μεγέθους, όπως καταλήξαμε έπειτα από διεξοδική αναζήτηση της υπάρχουσας βιβλιογραφίας. Στην συνέχεια, συζητείται το πρόβλημα του *αμυνόμενου-επιτιθέμενου* εντός δικτύων (Κεφάλαιο 3), μοντελοποιημένο ως POMDP. Τέλος παρουσιάζεται η ιδέα του *score function stochastic gradient* εκτιμητή (Κεφάλαιο 4), ο οποίος ανήκει στην οικογένεια των *policy-gradient* αλγορίθμων και με την κατάλληλη προσαρμογή, τροποποιείται, ώστε να εφαρμοστεί για την εύρεση μιας καλής *off-line global* πολιτικής, στο πλαίσιου το προβλήματος *αμυνόμενου-επιτιθέμενου*.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Λήψη αποφάσεων υπό αβεβαιότητα, POMDP

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Ενισχυτική μάθηση, αντιμετώπιση απειλών εντός δικτύων, στοχαστική βελτιστοποίηση, μπεϋζιανά γραφήματα εξάρτησης, προσεγγιστικοί αλγόριθμοι

ABSTRACT

For many years, the global scientific community is interested in methods regarding learning and decision making under uncertainty. Usually, there is an abundancy of data, the real word problem complexity is high and so is the desire on achieving the best performance. In that context, Markov Decision Processes have become a part of the puzzle, as they are very popular in the last three decades. Especially, POMDPs can model real world problems better than MDPs, with successful *robot navigation*, automated driving systems, voice recognition, finding a win strategy in games like chess, *Go* etc being examples of the previous case. Nevertheless, POMDPs have a major drawback, which is the fact that their optimal solution is practically impossible to be computed, especially regarding real world problems. This leads to the conclusion that the existing computational complexity makes the usage of approximate algorithms the one and only solution. Despite the fact that there is a plethora of algorithmic families that are able to solve (approximately) a POMDP problem, in this thesis we are focused on a particular *policy-gradient* method, which is derived from reinforcement learning. Generally speaking, the *policy-gradient* methods' idea is based on adapting the existing *decision-maker's* parameters suitably, in order to achieve the minimum average *long-term* cost. *Policy-gradient* methods seem extremely attractive, as they provide an approximation, which is feasible to compute, of big scale POMDPs' solutions.

At the beginning of this thesis, there is an analytical overview which starts from MDP's case (Chapter 1) and concludes to the POMDPs case (Chapter 2), which derives from the principles of the MDP's case. A detailed presentation of the basic concepts and methods, which are related to the POMDP model, is then presented, while at the end of Chapter 2, the so-called Holy Trinity of POMDPs' solving algorithms is highlighted. Those are able to cope with big scale POMDPs in our opinion, which was shaped after a thorough and lengthy search through the existing bibliography, as well as experience gained from experimenting on them. Continuing, the *attacker-defender problem* is discussed in the context of networks (Chapter 3), modelized as POMDP. Finally, the *score-function stochastic gradient estimator* is analyzed in Chapter 4. This estimator belongs to the policy-gradient algorithm family and with proper modification using suitable adopting, is able to find a good *off-line global* policy for the *defender-attacker* problem.

SUBJECT AREA: Decision making under uncertainty, POMDP

KEYWORDS: Reinforcement learning, addressing threats within networks, stochastic optimization, bayesian condition dependency graph, approximation algorithms

Στην οικογένεια μου & στην Ιφιγένεια...

ΕΥΧΑΡΙΣΤΙΕΣ

Κατ' αρχήν θεωρώ καθήκον μου να ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου κ. Καλουπτσίδη, για την αδιάλειπτη μέριμνα και στήριξη που μου παρέσχε, από την πρώτη κιόλας στιγμή. Πίστεψε σε εμένα αμέσως και με παρότρυνε συνεχώς για το καλύτερο. Κοντά του έμαθα πολλά πράγματα, πέραν του γεγονότος ότι αφύπνισε το ενδιαφέρον μου για τομείς των Μαθηματικών, των οποίων αγνοούσα την ύπαρξη. Αποτελούσε, αποτελεί και θα αποτελεί ένα σπουδαίο δάσκαλο και μέντορα, για όλους εκείνους που είχαν, έχουν και θα έχουν την τύχη να βρίσκονται δίπλα του.

Σε όλη αυτήν την προσπάθεια, καθοριστική ήταν, επίσης, η βοήθεια του υποψήφιου διδάκτορα Γιώργου Πικραμένου, τον οποίο οφείλω να ευχαριστήσω ιδιαιτέρως. Ευχαριστίες οφείλω, επιπλέον, και σε όλα τα υπόλοιπα μέλη της ομάδας, για την βοήθεια και κυρίως την υπομονή τους.

Ειδική μνεία αξίζει και στους φίλους μου, οι οποίοι με συντροφεύουν και κυρίως με ανέχονται όλα αυτά τα χρόνια. Καθένας τους μου έχει προσφέρει, με τον δικό του μοναδικό τρόπο, εμπειρίες ανεκτίμητης αξίας. Τους είμαι πραγματικά ευγνώμων.

Τέλος, ένα μεγάλο ευχαριστώ χρωστάω αδιαμφισβήτητα στην οικογένεια μου, τον πατέρα, την μητέρα αλλά και την γιαγιά μου, οι οποίοι είναι συνεχώς δίπλα μου, σε ότι και αν κάνω. Τους ευχαριστώ για όλα όσα μου έχουν προσφέρει, μέχρι σήμερα.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΠΡΟΛΟΓΟΣ

Με αυτή την πτυχιακή εργασία, κλείνει ενάς σημαντικός κύκλος της ζωής μου, ως προπτυχιακού φοιτητή. Όλα αυτά τα χρόνια, είχα την τύχη να συναντήσω καθηγητές αλλά και συμφοιτητές μου, τους οποίους θαυμάζω αφενός για τον χαρακτήρα τους και αφετέρου για τις διανοητικές τους ικανότητες. Μέσα σπό τις συζητήσεις των τελευταίων μηνών με τον επιβλέποντά καθηγητή μου, κ. Καλουπτσίδη, είχα την ευκαιρία να αντιληφθώ τι μου αρέσει πραγματικά και τι όχι, ενώ ταυτόχρονα συνειδητοποίησα πόσο ενδιαφέρον και ωραίο είναι να είσαι φοιτητής. Είμαι πραγματικά ευγνώμων, για όλα αυτά που συνέβησαν κατά την διάρκεια των προπτυχιακών μου σπουδών, αλλά κυρίως είμαι ευγνώμων στην δύναμη που με ενέπνεε συνεχώς να προχωρώ. Εύχομαι και ελπίζω τα βιώματα που αποκόμησα όλα αυτά τα χρόνια, να αποτελέσουν ορόσημο για το μέλλον.

1. ΕΙΣΑΓΩΓΗ

1.1 Κίνητρο

Εδώ και κάποιες δεκαετίες, δίνεται ιδιαίτερη σημασία στην λήψη αποφάσεων υπο αβεβαιότητα, με την βοήθεια διαφόρων τρόπων μάθησης και κατανόησης του εκάστοτε στοχαστικού περιβάλλοντος. Σε αυτήν την προσπάθεια, σημαντική είναι η συμβολή των (πλήρως παρατηρούμενων) Μαρκοβιανών διαδικασιών αποφάσεων (Markov Decision Processes - MDPs), καθώς και των μερικώς παρατηρούμενων MDPs (Partially Observed Markov Decision Processes - POMDPs). Αποδεικνύεται εμπράκτως ότι η χρήση Μαρκοβιανών διαδικασιών είναι ιδιαίτερα αποδοτική στην κατανόηση του υφιστάμενου περιβάλλοντος, με στόχο τον προσδιορισμό ενός όσο το δυνατόν καλύτερου τρόπου δράσης. Ειδικότερα, ας θεωρήσουμε την ύπαρξη ενός *decision-maker* (*agent*), ο οποίος αρχικά επιδιώκει να κατανοήσει το περιβάλλον στο οποίο βρίσκεται και ύστερα να δρά με τρόπο, όσο το δυνατό πλησιέστερο στον βέλτιστο. Σε περίπτωση όπου ο agent έχει την δυνατότητα να λαμβάνει αξιόπιστη πληροφορία σχετικά με την τρέχουσα κατάσταση (state) που βρίσκεται, το MDP μοντέλο είναι ιδιαίτερα πετυχημένο. Ώστοσο κάτι τέτοιο δεν καθίσταται πάντα εφικτό, με αποτέλεσμα η λαμβανόμενη πληροφορία να μην μπορεί να χρησιμοποιηθεί με τέτοιο άμεσο τρόπο. Επεκτείνοντας την ιδέα του MDP στο POMDP μοντέλο, ο agent αντιμετωπίζει την εισερχόμενη, από το περιβάλλον του, πληροφορία ως παρατήρηση της τρέχουσας κατάστασης (state) στην οποία βρίσκεται, αποδεχόμενος την ύπαρξη αβεβαιότητας κατά την διαδικασία της επεξεργασίας της. Γίνεται, επομένως, φανερό ότι στο πλαίσιο του POMDP μοντέλου, οι καταστάσεις του συστήματος δεν είναι παρατηρήσιμες, σε αντίθεση με τα MDPs.

Είναι διαισθητικά αντιληπτό ότι η εισαγωγή αβεβαιότητας, συνεπάγεται και την ραγδαία αύξηση του υπολογιστικού κόστους. Πράγματι έχει αποδειχθεί [19] ότι η επίλυση ενός POMDP πεπερασμένου ορίζοντα είναι ένα *PSPACE-Complete* πρόβλημα, θεωρώντας γνωστές τις παραμέτρους του μοντέλου. Ως επιβαρυντικός παράγοντας προστίθεται το γεγονός ότι συχνά κάτι τέτοιο δεν ισχύει σε προβλήματα του πραγματικού κόσμου. Προβάλλει, λοιπόν, επιτακτική η χρήση προσεγγιστικών αλγορίθμων, προκειμένου να υπερνικηθεί το έντονο υφιστάμενο υπολογιστικό κόστος.

Ο κύριος προσανατολισμός, ο οποίος επιλέχθηκε να ακολουθηθεί στο πλαίσιο της συγκεκριμένης εργασίας, είναι αποκλειστικά τα POMDPs, τα οποία χαρακτηρίζονται από ένα γενικό, ευέλικτο αλλά και κομψό πλαίσιο. Με αυτόν τον τρόπο, καθίσταται εφικτή η παρουσίαση της μοντελοποίησης ενός αυτοματοποιημένου μηχανισμού άμυνας δικτύων, κάτω από ρεαλιστικές υποθέσεις. Επιπλέον, γίνεται δυνατή η εκμετάλλευση ιδιαίτερα αποδοτικών και πρωτότυπων αλγορίθμων που προυπάρχουν για POMDPs, έχοντας ως στόχο την έυρεση της καλύτερης δυνατής λύσης για το πρόβλημα του *αμυνόμενου - επιτιθέμενου*, όπως εκείνο περιγράφεται στο τρίτο κεφάλαιο.

Στο υπόλοιπο αυτού του εισαγωγικού κεφαλαίου, θα γίνει μια σύντομη συζήτηση περί MDPs, τα οποία αποτελούν το σημείο εκκίνησης για την επέκταση βασικών εννοιών που χαρακτηρίζουν και τα POMDPs, τα οποία θα συζητηθούν αναλυτικά στο δεύτερο κεφάλαιο.

1.2 Μαρκοβιανές Διαδικασίες Αποφάσεων

Η Μαρκοβιανή διαδικασία δποφάσεων (MDP) αποτελεί Μαρκοβιανή διαδικασία (Markov Process - MP) με ανατροφοδοτούμενο έλεγχο. Ειδικότερα, ο *decision-maker*χρησιμοποιεί το state x_k της Μαρκοβιανής διαδικασίας κάθε χρονική στιγμή k, προκειμένου να λάβει μια ενέργεια u_k, η οποία με την σειρά της διαμορφώνει τον πίνακα μετάβασης, βάσει του οποίου θα προκύψει το επόμενο state x_{k+1}. Στόχος του *decision-maker*είναι να διαλέξει μια σειρά ενεργειών η οποία θα ελαχιστοποιεί το συσσωρευτικό κόστος που σχετίζεται με την αναμενόμενη τιμή της πορείας που θα ακολουθήσει η MP, κατά τον χρονικό ορίζοντα που έχει προκαθοριστεί.

1.2.1 MDP πεπερασμένου ορίζοντα

Το MDP πεπερασμένου ορίζοντα αποτελείται από τα ακόλουθα συστατικά στοιχεία:

- $\mathcal{X} = \{1, 2, ..., X\}$ υποδηλώνει τον χώρο καταστάσεων (state space),
- $\mathcal{U} = \{1, 2, ..., U\}$ υποδηλώνει τον χώρο ενεργειών (action space),
- Για κάθε u $\in \mathcal{U}$ και κάθε χρονική στιγμή $k \in \{0, 1, ..., N-1\}$, ο πίνακας μετάβασης του MDP, με βάση τον οποίο καθορίζεται το x_{k+1}, ειναι ο τετραγωνικός πίνακας με στοιχεία $P_{ij} = \mathbb{P}(x_{k+1} = j | x_k = i, u_k = u)$, $i, j \in \mathcal{X}$
- Για κάθε state $i \in \mathcal{X}$, ενέργεια $u \in \mathcal{U}$ και χρονική στιγμή $k \in \{0, 1, ..., N-1\}$ η ποσότητα $c(x, u, k)$ υποδηλώνει το κόστος ενός βήματος που ζημιώνεται ο decision-maker,
- Την χρονική στιγμή N, για κάθε state $i \in \mathcal{X}$, το $c_N(i)$ υποδηλώνει το τελικό κόστος.

Κάθε χρονική στιγμή k, ο *decision-maker*χρησιμοποιεί όλη την διαθέσιμη πληροφορία που έχει συγκεντρώσει ώς τότε, $H_k = \{x_0, u_0, ..., x_{k-1}, u_{k-1}, x_k\}$, με στόχο να επιλέξει μια ενέργεια $u_k = \pi_k(H_k)$, βασιζόμενος στην τρέχουσα πολιτική π_k. Το κριτήριο απόδοσης-αξιολόγησης των αποφάσεων του *decision-maker* είναι

$$J_{\boldsymbol{\pi}} = \mathbb{E}_{\boldsymbol{\pi}} \left\{ \sum_{i=0}^{N-1} c(x_k, \pi_k(H_k), k) + c_N(x_N) | x_0 = x \right\} \tag{1.1}$$

το οποίο αποτελεί το αναμενόμενο συσσωρευτικό κόστος που θα ζημιωθεί μέχρι την χρονική στιγμή N, ακολουθώντας την πολιτική $\boldsymbol{\pi} = \{\pi_0, ..., \pi_{N-1}\}$. Στόχος είναι η εύρεση της βέλτιστης πολιτικής

$$\arg\min_{\boldsymbol{\pi}} = J_{\boldsymbol{\pi}}(x) \tag{1.2}$$

Με άλλα λόγια, επιδιώκεται η εύρεση της πολιτικής $\boldsymbol{\pi}$ η οποία ελαχιστοποιεί το $J_{\boldsymbol{\pi}}(x)$, για κάθε αρχική κατάσταση $x_0 \in \mathcal{X}$. Δεδομένου μάλιστα ότι τα $\mathcal{X}$ και $\mathcal{U}$ είναι πεπερασμένα, το ελάχιστο υπάρχει πάντοτε, χωρίς απαραίτητα να είναι μοναδικό.

Προκειμένου να επιλυθεί το MDP για μια βέλτιστη πολιτική, είναι χρήσιμο σε αυτό το σημείο να εξετάσουμε τον χώρο των πολιτικών πιο αναλυτικά. Υπάρχουν οι ακόλουθοι τρείς διαφορετικοί τύποι πολιτικών:

- *Γενικές Πολιτικές*: Κάθε χρονική στιγμή k, ο *decision-maker* επιλέγει μια ενέργεια u_k σύμφωνα με την κατανομή $\pi_k(H_k)$. Συνεπώς η u_k είναι μια πιθανοτική συνάρτηση του H_k,
- *Τυχαίες Μαρκοβιανές Πολιτικές*: Η ενέργεια u_k επιλέγεται σύμφωνα με την κατανομή $\pi_k(x_k)$ και προφανώς αποτελεί πιθανοτική συνάρτηση του state x_k μόνο,
- *Ντετερμινιστικές Μαρκοβιανές Πολιτικές*: Η ενέργεια u_k επιλέγεται ντετερμινιστικά με βάση ένα mapping από τον χώρο καταστάσεων στον χώρο ενεργειών.

Το βασικότερο χαρακτηριστικό του MDP προβλήματος (1.2) είναι το γεγονός ότι για την επίτευξη του ελαχίστου, αρκεί να ληφθούν υπόψη μόνο οι ντιτερμινιστικές Μαρκοβιανές πολιτικές. Ως συνέπεια η $J_{\boldsymbol{\pi}}$ της (1.1) υπολογίζεται μόνο ως προς $\{x_0, x_1, ..., x_N\}$, μιας και η u_k είναι ντετερμινιστική συνάρτηση του x_k.

Ακολούθως παρουσιάζεται ένα ιδιαίτερο σημαντικό αποτέλεσμα που συνδέει τον Δυναμικό Προγραμματισμό με την εύρεση της βέλτιστης πολιτικής του MDP προβλήματος (1.1) - (1.2). Ειδικότερα η βέλτιστη πολιτική $\boldsymbol{\pi}^*$ μπορεί να εντοπιστεί με την βοήθεια του στοχαστικού αλγορίθμου δυναμικού προγραμματισμού Bellman [1], όπως διαπιστώνεται και από το *Θεώρημα 1.1*.

Θεώρημα 1.1 (Αλγόριθμος Δυναμικού Προγραμματισμού Bellman): Η βέλτιστη πολιτική $\boldsymbol{\pi}^*$ για το πεπερασμένου ορίζοντα MDP της παρούσας ενότητας, μπορεί να προσδιοριστεί πλήρως από την λύση του ακόλουθου προβλήματος:

$$J_N(i) = c(i, N), \forall i \in \mathcal{X} \tag{1.3}$$

$$J_k(i) = \min_{u \in \mathcal{U}} \left\{ c(i, u, k) + \sum_j P_{ij}(u, k) J_{k+1}(j) \right\}, \tag{1.4}$$

$$\pi_k^* = \arg\min_{u \in \mathcal{U}} \left\{ c(i, u, k) + \sum_j P_{ij}(u, k) J_{k+1}(j) \right\} \tag{1.5}$$

Απόδειξη: [14, σελ. 125]

Αποδεικνύεται, επίσης, ότι η ντιντερμινιστική Μαρκοβιανή πολιτική $\boldsymbol{\pi}^*$ που προκύπτει από την επίλυση του προβλήματος (1.3) - (1.5) είναι βέλτιστη, δηλαδή ικανοποιεί την (1.2). Αντιθέτως εάν μια ντετερμινιστική Μαρκοβιανή πολιτική $\boldsymbol{\pi}^*$ ικανοποιεί την (1.2), τότε αποτελεί και λύση του προβλήματος (1.3) - (1.5).

Σε ορισμένες περιπτώσεις όμως, είναι προτιμότερο η εξίσωση Bellman να εκφραστεί υπό την μορφή forward αναδρομής. Χαρακτηριστική είναι η περίπτωση του MDP άπειρου χρονικού ορίζοντα, η οποία θα μελετηθεί συνοπτικά στην ακόλουθη υποενότητα. Όπως είναι εκφρασμένος ο Αλγόριθμος Δυναμικού Προγραμματισμού Bellman στις (1.3) - (1.5), είναι προφανές ότι πρόκειται για μια backward αναδρομή, η οποία αρχικοποιείται την χρονική στιγμή $k = 0$ με J_N και εξελίσσεται προς τα πίσω. Με στόχο, λοιπόν, να αναδιαμορφώσουμε την εξίσωση Bellman (1.4), ορίζουμε ως value function

$$V_n(x) = J_{N-n}(x), 0 \leq n \leq N, x \in \mathcal{X} \tag{1.6}$$

Από την (1.4) έπεται ότι η $V_n(x)$ ικανοποιεί την ακόλουθη εξίσωση δυναμικού προγραμματισμού

$$V_n(i) = \min_{u \in \mathcal{U}} \left\{ c(i, u, N-n) + \sum_j P_{ij}(u, N-n) V_{n-1}(j) \right\} \tag{1.7}$$

1.2.2 MDP discounted κόστους με άπειρου ορίζοντα

Επεκτείνοντας την ιδέα του MDP πεπερασμένου ορίζοντα για $N \to \infty$, οδηγούμαστε στο MDP άπειρου ορίζοντα. Όπως είναι αναμενόμενο, δεν είναι δυνατό, πλέον, να θεωρούνται οι πιθανότητες μετάβασης καθώς και το κόστος με το οποίο ζημιώνεται ο *decision-maker*, τόσο ξεκάθαρα εξαρτώμενα από τον χρόνο, ενώ ταυτόχρονα είναι φανερό πως δεν υφίσταται πια τερματικό κόστος.

Παρομοίως με την υποενότητα 1.2.1, στόχος είναι η εύρεση μιας βέλτιστης πολιτικής $\boldsymbol{\pi} = \arg\min_{\boldsymbol{\pi}} J_{\boldsymbol{\pi}}(i)$, όπου το $J_{\boldsymbol{\pi}}$ υποδηλώνει το συσσωρευτικό discounted κόστος που ζημιώνεται ο *decision-maker*, σε βάθος άπειρου χρονικού ορίζοντα

$$J_{\boldsymbol{\pi}}(i) = \mathbb{E}_{\boldsymbol{\pi}} \left\{ \sum_{k=0}^{\infty} \rho^k c(x_k, u_k) | x_0 = i \right\} \tag{1.8}$$

με $\rho \in [0, 1)$: discount παράγοντας που σταθμίζει το κόστος κάθε χρονική στιγμή k. Είναι προφανές ότι οι αρχικές αποφάσεις επηρεάζουν εντονότερα το συσσωρευτικό κόστος, σε αντίθεση με μεταγενέστερες. Επιπρόσθετα, ο discount παράγοντας ρ εξασφαλίζει ότι το $J_{\boldsymbol{\pi}}(i)$ είναι πάντοτε φραγμένο, καθώς ισχύει

$$J_{\boldsymbol{\pi}}(i) \leq \max_{i \in \mathcal{X}, u \in \mathcal{U}} \frac{|c(i, u)|}{1 - \rho} \tag{1.9}$$

Συμπληρωματικά με τους τύπους των πολιτικών που ορίστηκαν στο πλαίσιο του MDP πεπερασμένου ορίζοντα, είναι αναγκαίο σε αυτό το σημείο να προστεθεί η έννοια της *στάσιμης πολιτικής*, καθώς κατέχει ιδιαίτερη σημασία για τα MDP άπειρου ορίζοντα (βλ. *Θεώρημα 1.2*). Για μία *στάσιμη πολιτική* ισχύει ότι $\boldsymbol{\pi} = \{\pi, \pi, ..., \pi\}$, πράγμα που υποδηλώνει ότι δεν μεταβάλλεται με την πάροδο του χρόνου.

Θεωρώντας ένα MDP άπειρου ορίζοντα, ως ένα MDP πεπερασμένου ορίζοντα με $N \to \infty$ και βασιζόμενοι στις (1.6) - (1.7), είναι επιθυμητό να δοθεί η εξίσωση δυναμικού προγραμματισμού. Στην συγκεκριμένη περίπτωση είναι βολικότερο να οριστεί η value function ως

$$V_n(i) = \rho^{n-N} J_{N-n}(i), 0 \leq n \leq N, i \in \mathcal{X} \tag{1.10}$$

που ικανοποιεί την εξίσωση δυναμικού προγραμματισμού

$$V_n(i) = \min_{u \in \mathcal{U}} \left\{ c(i, u) + \rho \sum_j P_{ij}(u) V_{n-1}(j) \right\}, V_0(i) = 0 \tag{1.11}$$

Προφανώς η εξίσωση δυναμικού προγραμματισμού για την περίπτωση των MDP άπειρου ορίζοντα, προκύπτει καθώς το $N \to \infty$ στην (1.11). Το ακόλουθο θεώρημα θεμελιώνει τρείς βασικούς ισχυρισμούς για τα MDP άπειρου ορίζοντα

Θεώρημα 1.2: Ας θεωρηθεί ένα άπειρου ορίζοντα discounted κόστους MDP, με παράγοντα ρ. Τότε:

- Για κάθε αρχική κατάσταση i, το βέλτιστο συσσωρευτικό κόστος J_π^* επιτυγχάνεται από την value function $V(i)$, που ικανοποιεί την εξίσωση Bellman (1.12) - (1.13),
- Για κάθε αρχική κατάσταση i, το βέλτιστο συσσωρευτικό κόστος J_π^* επιτυγχάνεται από την στάσιμη ντετερμινιστική Μαρκοβιανή πολιτική π^*, που ικανοποιεί την εξίσωση Bellman (1.12) - (1.13),
- Η value function V είναι η μοναδική λύση της εξίσωσης Bellman (1.12) - (1.13).

$$V(i) = \min_{u \in \mathcal{U}} Q(i,u),\ Q(i,u) = c(i,u) + \rho \sum_j P_{ij}(u) V(j), \tag{1.12}$$

$$\pi^* = \arg\min_{u \in \mathcal{U}} Q(i,u) \tag{1.13}$$

Απόδειξη: [14, σελ. 130]

1.2.3 Αριθμητικές Μέθοδοι Επίλησης MDP discounted κόστους άπειρου ορίζοντα

Αυτή η υποενότητα πραγματεύεται τις τρείς πιο κλασικές μεθόδους επίλυσης του MDP άπειρου ορίζοντα με discounted κόστος. Ειδικότερα πρόκειται για την *Value Interation (VI) Μέθοδο*, την *Policy Interation (PI) Μέθοδο* καθώς και με την χρήση *Γραμικού Προγραμματισμού (LP)*. Για πιο δεξοδική μελέτη αυτών των μεθόδων, αρκετά καλές πηγές αποτελούν τα [4, 6, 27].

Value Interation (VI) Μέθοδος

Με την *VI μέθοδο* επιτυγχάνεται η εύρεση μιας αρκετά καλής προσέγγισης της value function V, η οποία ικανοποιεί τις (1.12) - (1.13). Επιλέγοντας έναν ικανοποιητικά μεγάλο αριθμό επαναλήψεων N και αρχικοποιώντας $V_0(i) = 0, \forall i \in \mathcal{X}$ υπολογίζεται επαναληπτικά

$$V(i) = \min_{u \in \mathcal{U}} Q_k(i,u), \pi_k^* = \arg\min_{u \in \mathcal{U}} Q_k(i,u), \tag{1.14}$$

$$Q_k(i,u) = c(i,u) + \rho \sum_j P_{ij}(u) V_{k-1}(j) \tag{1.15}$$

Η *VI μέθοδος* δίνεται υπό την μορφή ψευδογλώσσας στον **Αλγόριθμο 1**.

Algorithm 1 Value Interation Μέθοδος (1.14 - 1.15)

1: Αρχικοποιήστε $V_0(i) \leftarrow 0, \forall i \in \mathcal{X}$
2: **for** $k = 1$ to N **do**
3: **for each** $i \in \mathcal{X}$ **do**
4: **for each** $u \in \mathcal{U}$ **do**
5: $Q(i,u) \leftarrow c(i,u) + \rho \sum_j P_{ij}(u) V_{k-1}(j)$
6: **end for**
7: $V_k(i) \leftarrow \min_{u \in \mathcal{U}} Q(i,u)$
8: $\pi_k(i) \leftarrow \arg\min_{u \in \mathcal{U}} V_k(i)$
9: **end for**
10: **end for**
11: **return** $\pi_N(i), \forall i \in \mathcal{X}$

Έπειτα χρησιμοποιείται η στάσιμη πολιτική π_N κάθε χρονική στιγμή $k = 1, 2, \ldots$. Είναι φανερό ότι ο αλγόριθμος της Value Interation μεθόδου που παρουσιάστηκε, είναι πανομοιότυπος με την εξίσωση Bellman, η οποία αντιστοιχεί σε MDP πεπερασμένου ορίζοντα

(1.11). Μοναδική διαφορά αποτελεί το γεγονός ότι στην περίπτωση του MDP άπειρου ορίζοντα, τελικά, ακολουθείται μια στάσιμη πολιτική. Τροποποιώντας ελαφρώς τον *Αλγόριθμο 1*, μπορεί να προκύψει πολύ εύκολα η εκδοχή της *VI μεθόδου* για την περίπτωση των MDP πεπερασμένου ορίζοντα.

Όσον αφορά την σύγκλιση της *VI μεθόδου*, το ακόλουθο θεώρημα μας διαβεβαιώνει, ότι συγκλίνει γεωμετρικά στην βέλτιστη value function V^* που ικανόποιει τις (1.12) - (1.13). Ειδικότερα

Θεώρημα 1.3 (Σύγκλιση μεθόδου Value Interation): Η *μέθοδος VI* συγκλίνει γεωμετρικά στη βέλτιστη value function που ικανόποιει τις (1.12) - (1.13), δηλαδή

$$|V(i) - V_k(i)| \leq \frac{\rho^{k+1}}{1-\rho} \max_{i \in \mathcal{X}, u \in \mathcal{U}} |c(i,u)|$$

Απόδειξη: [14, σελ. 130]

Τέλος, πέραν της κλασικής *VI μεθόδου* που παρουσιάστηκε σε αυτήν την υποενότητα, υπάρχει και η *Ασύχρονη VI μέθοδος*, που αποτελεί την παραλληλοποιημένη μορφή της πρώτης. Η βασική ιδέα της είναι, ότι ένας processor μπορεί να διαθέτει ένα διάνυσμα $V(i)$ και επιλέγει να εφαρμόσει την κλασική *VI μέθοδο* για κάποιο συγκεκριμένο μόνο $i \in \mathcal{X}$, αυθαίρετο πλήθος φορών. Εν συνέχεια ενημερώνει τους υπόλοιπους precessors για το $V(i)$ στο οποίο κατέληξε και εκείνοι με την σειρά τους, το υιοθετούν. Κάτω από ορισμένες συνθήκες, αποδεικνύεται ότι και η *Ασύγχρονη VI μέθοδος* συγκλίνει στην V^*. Για πιο εκτενή ανάλυση της μέθοδου, ο αναγνώστης παραπέμπεται [2, 4, 6].

Policy Interation (PI) Μέθοδος

Η *PI μέθοδος* αποτελεί έναν επαναληπτικό αλγόριθμο, ο οποίος υπολογίζει και βελτιώνει την υπάρχουσα πολιτική σε κάθε επανάληψη. Δεδομένου ότι οι χώροι ενεργειών και καταστάσεων είναι πεπερασμένοι, είναι φανερό ότι το πλήθος των στάσιμων πολιτικών είναι πεπερασμένο, και συγκεκριμένα $|\mathcal{X}|^{|\mathcal{U}|}$. Κατά συνέπεια, μια naive προσέγγιση υλοποίησης της *PI μεθόδου* είναι η επάναληψη πάνω από πεπερασμένο πλήθος πολιτικών, χρησιμοποιώντας την (1.12) προκειμένου να υπολογιστούν οι αντίστοιχες value functions και εν τέλει να επιλέγει εκείνη με την μικρότερη τιμή. Προφανώς κάτι τέτοιο είναι ιδιαίτερα αναποτελεσματικό, εξαιτίας της ύπαρξης πεπερασμένου μεν, αλλά εκθετικού αριθμού στάσιμων πολιτικών.

Ειδικότερα, η μέθοδος αποτελείται από δύο κύριες φάσεις

- *Βελτίωση της τρέχουσας πολιτικής*, όπου πραγματοποιείται ο υπολογισμός της π_k πολιτικής, θεωρώντας δεδομένη την στάσιμη πολιτική π_{k-1} της προηγούμενης επανάληψης, η οποία σχετίζεται με το συσσωρευτικό κόστος $J_{\pi_{k-1}}$. Η νέα πολιτική υπολογίζεται ως εξής

$$\pi_k(i) = \arg\min_{u \in \mathcal{U}} \left\{ c(i,u) + \rho \sum_j P_{ij}(u) J_{\pi_{k-1}}(i) \right\}, \forall i \in \mathcal{X} \tag{1.16}$$

- *Αξιολόγηση της προκύπτουσας πολιτικής*, όπου δοθείσης πλέον της νέας πολιτικής που προέκυψε, υπολογίζεται το discounted συσσωρευτικό κόστος που σχετίζεται με την πολιτικής ως εξής

$$J_{\pi_k}(i) = c(i, \pi_k(i)) + \rho \sum_j P_{ij}(u) J_{\pi_k}(i), \forall i \in \mathcal{X} \tag{1.17}$$

το οποίο αποτελεί ένα γραμμικό σύστημα ως προς το J_{π_k} και μπορεί να επιλυθεί $\forall i \in \mathcal{X}$.[1]

Η *PI μέθοδος* δίνεται υπό την μορφή ψευδοκώδικα στον **Αλγόριθμο 2**.

Είναι εύκολο να δειχτεί ότι με την εφαρμογή του αλγορίθμου, δημιουργείται μια σειρά πολιτικών, οι οποίες μονότονα μειώνονται ως προς το συσχετιζόμενο με εκείνες συσσωρευτικό κόστος $J_{\pi_k}, k = 0, 1, 2, ...$.

Algorithm 2 Policy Interation Μέθοδος (1.16 - 1.17)

$k \leftarrow 0$
Επιλέξτε $\pi_0(i), \forall i \in \mathcal{X}$
Επιλύστε το γραμμικό σύστημα $J_{\pi_0}(i) = c(i, \pi_0(i)) + \rho \sum_j Pij(\pi_0(i)) J_{\pi_0}(i), \forall i \in \mathcal{X}$
do
  $k \leftarrow k + 1$
  for each $i \in \mathcal{X}$ **do**
    for each $u \in \mathcal{U}$ **do**
      $Q(i, u) \leftarrow c(i, u) + \rho \sum_j P_{ij}(u) J_{\pi_{k-1}}$
    end for
    $\pi_k(i) = \arg\min_{u \in \mathcal{U}} Q(i, u)$
  end for
  Επιλύστε το γραμμικό σύστημα $J_{\pi_k}(i) = c(i, \pi_k(i)) + \rho \sum_j P_{ij}(\pi_k(i)) J_{\pi_k(i)}, \forall i \in \mathcal{X}$
while $J_{\pi_k}(i) < J_{\pi_{k-1}}(i), \forall i \in \mathcal{X}$
return $\pi_k(i), \forall i \in \mathcal{X}$

Πέραν της κλασικής *PI μεθόδου* που περιγράφηκε, παρομοίως με την *VI μέθοδο*, υπάρχει και η *ασύγχρονη* εκδοχή της. Επιπλέον είναι δυνατή η τροποίηση της κλασικής μεθόδου, ώστε να μην λαμβάνεται υπόψη, στο βήμα βελτίωσης της πολιτικής, μόνο εκείνη του προηγούμενου βήματος, αλλά των m προγενέστερων βημάτων. Κάτι τέτοιο επιτυγχάνεται με την βοήθεια δυναμικού προγραμμάτισμου $m-$βημάτων. Για λεπτομερέστερη ανάλυση επί των συγκεκριμένων θεμάτων, ο αναγνώστης παραπέμπεται [2, 3, 4].

Γραμμικός Προγραμματισμός (LP)

Έστω ότι χρησιμοποιείται η *VI μέθοδος*, προκειμένου να δημιουργηθούν μια σειρά από διανύσματα $V_k(i), \forall i \in \mathcal{X}$, ξεκινώντας από ένα αρχικό διάνυσμα $V_0(i), \forall i \in \mathcal{X}$, τέτοιο, ώστε

$$V_0(i) \leq \min_{u \in \mathcal{U}} \left\{ c(i, u) + \rho \sum_j P_{ij}(u) V_0(j) \right\} \tag{1.18}$$

Ως συνέπεια, θα ισχύει $V_k(i) \leq V_{k+1}(i), \forall k = 0, 1, ...$, λόγω της ιδιότητας της μονοτονίας που χαρακτηρίζει τον δυναμικό προγραμματισμό (βλ. [2] Κεφάλαιο 1, Ασκ. 1.23). Ταυτόχρονα, καθώς το $k \to \infty$, γνωρίζουμε ότι $V_k \to V^*$. Επομένως συνδυάζοντας αυτά τα δύο επιχειρήματα προκύπτει ότι $V_0(i) \leq V^*(i), \forall i \in \mathcal{X}$. Είναι προφανές, λοιπόν, ότι η V^* είναι η "μεγαλύτερη" V που ικανοποιεί τον περιορισμό

$$V(i) \leq c(i, u) + \rho \sum_j P_{ij}(u) V(j), \forall i \in \mathcal{X}, u \in \mathcal{U} \tag{1.19}$$

[1] Προφανώς κλασικές μέθοδοι επίλυσης γραμμικών συστημάτων, όπως για παράδειγμα η μέθοδος Απαλοιφής του Gauss, δεν αποτελούν ελκυστικές επιλογές. Σε αντίθεση, αρκετές φορές είναι πιο αποδοτικό να προσεγγιστεί το βήμα αξιολόγησης της προκύπτουσας πολιτικής, με μερικές επαναλήψεις της *VI μεθόδου*.

Επομένως, η V^* είναι η λύση του ακόλουθο προβλήματος γραμμικού προγραμματισμού (1.20) - (1.21):

$$\max_V \sum_i V(i) \tag{1.20}$$

$$\text{s.t. } V(i) \leq c(i,u) + \rho \sum_j P_{ij}(u)V(i), i \in \mathcal{X}, u \in \mathcal{U} \tag{1.21}$$

Δυστυχώς για χώρους καταστάσεων μεγάλου μεγέθους, το πρόβλημα γραμμικού προγραμματισμού αυξάνεται δραματικά και η επίλυση του γίνεται πλέον μη πρακτική. Ενδεικτικά η μέθοδος Simplex, η οποία αποτελεί την πιο δημοφιλή επιλογή για την επίλυση προβλημάτων γραμμικού προγραμματισμού, επιβαρύνεται εκθετικά κατά την αύξηση της διάστασης του προβλήματος. Γίνεται πλέον εμφανές ότι σε προβλήματα του πραγματικού κόσμου αποφεύγεται η ακριβής επίλυση του ανωτέρω γραμμικού προβλήματος. Αντιθέτως, είναι ελκυστικότερη η επίλυση κατά προσέγγιση προβλημάτων γραμικού προγραμματισμού, με στόχο των εντοπισμό προσεγγιστικών λύσεων του αρχικού προβλήματος δυναμικού προγραμματισμού ([4]).

1.2.4 MDP μέσου κόστους άπειρου ορίζοντα

Σε αντίθεση με το discounted κόστος, το μέσο κόστος αντιστοιχεί στην ποσότητα που ζημιώνεται ο *decision-maker* κάθε χρονική στιγμή, καθώς ο χρόνος έχει κυλίσει αρκετά ($N \to \infty$). Ως συνέπεια, οι αποφάσεις που λαμβάνονται στην αρχή δεν παίζουν τόσο καθαριστικό ρόλο, όπως αντιθέτως συμβαίνει στην περίπτωση του MDP discounted κόστους.

Ας θεωρήσουμε μια στάσιμη κατανομή π και ας ορίσουμε το άπειρου ορίζοντα μέσο κόστος ως

$$J_\pi(x_0) = \lim_{N\to\infty} \frac{1}{N+1} \mathbb{E}_\pi \left\{ \sum_{k=0}^{N} c(x_k, u_k) | x_0 \right\} \tag{1.22}$$

Ως βέλτιστη πολιτική π^* θεωρείται εκείνη για την οποία ισχύει $J^*_\pi(x_0) \leq J_\pi(x_0), \forall \pi \in \Pi, \forall x_0 \in \mathcal{X}$, όπου Π το σύνολο όλων των διαθέσιμων στάσιμων κατανομών.

Σε αντίθεση με τα προβλήματα discounted κόστους, εδώ η δομή του πίνακα μετάβασης $P(u)$ έχει ιδιαίτερα σημαντικό ρόλο, καθώς καθορίζει την ύπαρξη ή όχι μιας βέλτιστης στάσιμης κατανομής. Η περαιτέρω ανάλυση επ' αυτού, υπερβαίνει τον σκοπό της συγκεκριμένης εργασίας. Θα κινηθούμε αρκετά "συντηρητικά" ως προς αυτό το θέμα, μένοντας περιορισμένοι σε MDP με πεπερασμένο αριθμό καταστάσεων. Ταυτόχρονα εισάγεται η έννοια του *unichain MDP*, η οποία είναι επαρκής για τις ανάγκες της συγκεκριμένης εργασίας. Ειδικότερα:

Ορισμός: Ένα MDP λέγεται *unichain* εάν κάθε Ντερμινιστική στάσιμη πολιτική αντιστοιχεί σε μία Μαρκοβιανή αλυσίδα με μία μόνο *recurrent κλάση*.

Προκύπτει ότι, εάν ένα MDP μέσου κόστους είναι *unichain*, τότε υπάρχει πάντοτε κάποια βέλτιστη στάσιμη πολιτική. Το ακόλουθο θεώρημα είναι ανάλογο της εξίσωσης δυναμικού προγραμματισμού Bellman

Θεώρημα 1.4: Θεωρώντας πεπερασμένο χώρο ενεργειών και καταστάσεων και ταυτόχρονα υποθέτοντας ότι το MDP είναι *unichain* μέσου κόστους, τότε είναι βέβαιο ότι υπάρχει

κάποια βέλτιστη στάσιμη ντετερμινιστική πολιτική π^* η οποία ικανοποιεί

$$g + V(i) = \min_{u \in \mathcal{U}} \left\{ c(i,u) + \sum_j P_{ij}(u) V(j) \right\} \tag{1.23}$$

$$\pi^*(i) = \min_{u \in \mathcal{U}} \left\{ c(i,u) + \sum_j P_{ij}(u) V(j) \right\} \tag{1.24}$$

με $g = J_{\pi^*}(x_0)$, όπου $J_{\pi^*}(x_0)$ υποδηλώνει το μέσο κόστος που ζημιώνεται ο *decision-maker* επιλέγοντας να ακολουθήσει την βέλτιστη πολιτική π^* και είναι ανεξάρτητο από την αρχική κατάσταση x_0.

Απόδειξη: [14, σελ. 133]

1.2.5 Αριθμητικές Μέθοδοι Επίλυσης MDP μέσου κόστους άπειρου ορίζοντα

Αυτή η υποενότητα ασχολείται με τις δύο βασικές μεθόδους επίλυσης του MDP άπειρου ορίζοντα μέσου κόστους. Ειδικότερα πρόκειται για την *Relative Value Interation (RVI) Μέθοδο*, καθώς και την χρήση *Γραμικού Προγραμματισμού (LP)*.

Relative Value Interation (RVI) Μέθοδος

Με σημείο αφετηρίας την κύρια ιδέα της *VI μεθόδου*, θα περιγραφεί ακολούθως η *RVI μέθοδος*, η οποία αποτελεί μια παραλλαγή της *VI μεθόδου*, για MDPs μέσου κόστους με άπειρο ορίζοντα.

Επιλέγοντας ένα συγκεκριμένο $i_0 \in \mathcal{X}$, ορίζουμε την $\tilde{V}(i) = V(i) - V(i_0)$. Προφανώς ισχύει $\tilde{V}(i_0) = 0$, ενώ επίσης είναι προφανές ότι η $\tilde{V}(i), \forall i \in \mathcal{X}$ ικανοποιεί την (1.23). Συνεπώς προκύπτει

$$g = \min_{u \in \mathcal{U}} \left\{ c(1,u) + \sum_j P_{1j}(u) \tilde{V}(j) \right\}, i = 1 \tag{1.25}$$

$$g + \tilde{V}(i) = \min_{u \in \mathcal{U}} \left\{ c(i,u) + \sum_j P_{ij}(u) \tilde{V}(j) \right\}, i > 1 \tag{1.26}$$

Παρομοίως με την *VI μέθοδο*, οι (1.25-1.26) συνθέτουν την *RVI μέθοδο*, η οποία για κάθε $k = 0, 1, 2, ...$

$$g_k = \min_{u \in \mathcal{U}} \left\{ c(1,u) + \sum_j P_{1j}(u) \tilde{V}_{k-1}(j) \right\} \tag{1.27}$$

$$\tilde{V}_k(i) = \min_{u \in \mathcal{U}} \left\{ c(i,u) + \sum_{j>1} P_{ij}(u) \tilde{V}_{k-1}(j) \right\} - g_k \tag{1.28}$$

Η *PVI μέθοδος* δίνεται σε μορφή ψευδογλώσσας στον **Αλγόριθμο 3**.

Γραμμικός Προγραμματισμός (LP)

Θεωρώντας $\pi(x,u) = \mathbb{P}(x_k = x, u_k = u), x \in \mathcal{X}, u \in \mathcal{U}$ την από κοινού πιθανότητα κατάστασης ενέργειας, το ακόλουθο θεώρημα διατυπώνει το πρόβλημα εύρεσης της βέλτιστης πολιτικής για MDPs μέσου κόστους με άπειρο ορίζοντα, ως πρόβλημα γραμμικού προγραμματισμού. Ειδικότερα:

Algorithm 3 Relative Value Interation Μέθοδος (1.25 - 1.28)

Αρχικοποιήστε $\tilde{V}_0(i) \leftarrow 0, \forall i \in X$
for $k = 1$ to N **do**
  for each $u \in \mathcal{U}$ **do**
    $g_k(u) \leftarrow c(1,u) + \sum_j P_{1j}(u)\tilde{V}_{k-1}(j)$
  end for
  $g_k \leftarrow \min_{u\in\mathcal{U}} \{g_k(u)\}$
  for each $i \in \mathcal{X} - \{1\}$ **do**
    for each $u \in \mathcal{U}$ **do**
      $Q_k(i,u) \leftarrow c(i,u) + \sum_{j>1} P_{ij}(u)\tilde{V}_{k-1}(j)$
    end for
    $\tilde{V}_k(i) \leftarrow \min_{u\in\mathcal{U}}\{Q_k(i,u)\} - g_k$
  end for
end for
return $\tilde{V}_N(i), \forall i \in \mathcal{X}$

Θεώρημα 1.5: Θεωρώντας ένα MDP με πεπερασμένο αριθμό καταστάσεων και ενεργειών, το οποίο ταυτόχρονα είναι και *unichain* μέσου κόστους MDP, προκύπτει ότι η βέλτιστη πολιτική π^* είναι

$$\pi^*(x) = u \text{ με πιθανότητα } \theta_{x,u} = \mathbb{P}(action = u | state = x) = \frac{\pi^*(x,u)}{\sum_{i\in\mathcal{X}} \pi^*(i,u)}, x \in \mathcal{X}, u \in \mathcal{U} \tag{1.29}$$

Τα $|\mathcal{X}|\times|\mathcal{U}|$ στοιχεία του π^* είναι η λύση του ακόλουθου προβλήματος γραμμικού προγραμματισμού

$$\min_{\pi} \sum_{i\in\mathcal{X}} \sum_{u\in\mathcal{U}} c(i,u)\pi(i,u) \tag{1.30}$$

$$\text{s.t. } \pi(i,u) \geq 0, \forall i \in \mathcal{X}, u \in \mathcal{U}, \sum_{i\in\mathcal{X}} \sum_{u\in\mathcal{U}} \pi(i,u) = 1 \tag{1.31}$$

$$\sum_{u\in\mathcal{U}} \pi(j,u) = \sum_{i\in\mathcal{X}} \sum_{u\in\mathcal{U}} \pi(i,u)P_{ij}(u), \forall j \in \mathcal{X} \tag{1.32}$$

Απόδειξη: [14, σελ. 136]

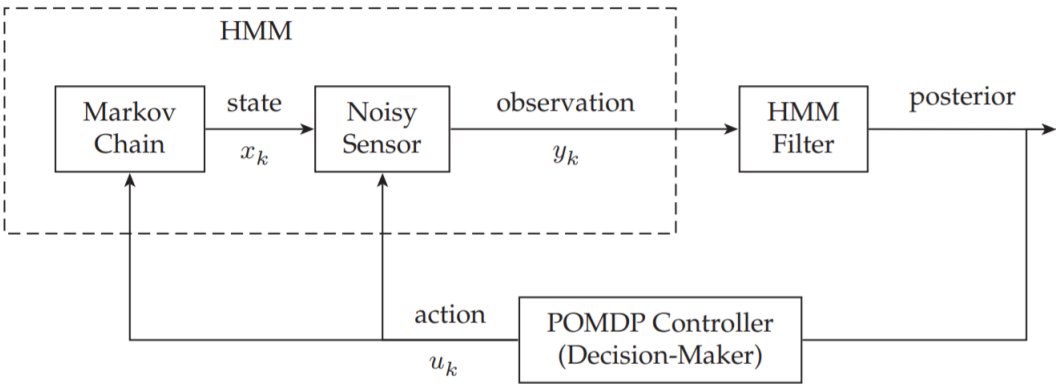
Μπορεί το *Θεώρημα 1.5* να υπονοεί ότι η π^* είναι randomized πολιτική, στην πραγματικότητα, όμως, είναι ντετερμινιστική πολιτική με $\pi(x,u) > 0$ μόνο όταν $u = \pi^*(i)$, διαφορετικά $\pi(x,u) = 0$.

2. ΜΕΡΙΚΩΣ ΠΑΡΑΤΗΡΟΥΜΕΝΕΣ ΜΑΡΚΟΒΙΑΝΕΣ ΔΙΑΔΙΚΑΣΙΕΣ ΑΠΟΦΑΣΕΩΝ

2.1 Βασικά στοιχειά του POMDP μοντέλου

2.1.1 Εισαγωγή

Οι μερικώς παρατηρούμενες Μαρκοβιανές διαδικασίες αποφάσεων (POMDP) αποτελούν επέκταση των κρυφών Μαρκοβιανών μοντέλων (Hidden Markov Models - HMM). Τα HMM αποτελούνται από μια αλυσίδα Markov $\{x_k\}$ με $|\mathcal{X}|$ καταστάσεις, οι οποίες όμως δεν είναι άμεσα παρατηρήσιμες. Η υφιστάμενη Μαρκοβιανή αλυσίδα παρατηρείται με την βοήθεια θορυβωδών παρατηρήσεων $\{y_k\}$. Διαφωτιστικό είναι το *Σχήμα 2.1*, η οποία παρουσιάζει σχηματικά την δομή ενός κλασικού POMDP, όπου η λαμβανόμενη ενέργεια u_k κάθε χρονική στιγμή, επηρεάζει την επόμενη κατάσταση ή/και την επόμενη παρατήρηση του υφιστάμενου HMM. Το HMM Φίλτρο (βλ. *Παράρτημα Ι*) υπολογίζει την posterior κατανομή



Σχήμα 2.1: Σχηματικά η δομή ενός κλασικού POMDP

b_k, στην προσπάθεια του να εκτιμήσει το x_k, δοθείσης της παρατήρησης y_k, κάθε χρονική στιγμή k. Η posterior κατανομή b_k καλείται κατάσταση belief (βλ. 2.1.3).

Στις ακόλουθες υποενότητες περιγράφεται αναλυτικότερα το POMDP μοντέλο καθώς και οι βασικές αρχές που το διέπουν, ξεκινώντας από την πιο απλή εκδοχή του, το POMDP μοντέλο πεπερασμένου ορίζοντα.

2.1.2 POMDP πεπερασμένου ορίζοντα

Το POMDP πεπερασμένου ορίζοντα αποτελείται από τα ακόλουθα συστατικά στοιχεία:

- $\mathcal{X} = \{1, 2, ..., X\}$ υποδηλώνει τον χώρο καταστάσεων (state space),
- $\mathcal{U} = \{1, 2, ..., U\}$ υποδηλώνει τον χώρο ενεργειών (action space),
- $\mathcal{Y} = \{1, 2, ..., Y\}$ υποδηλώνει τον χώρο παρατηρήσεων (observation space),
- Για κάθε u $\in \mathcal{U}$ και κάθε χρονική στιγμή k $\in \{0, 1, ..., N-1\}$, ο πίνακας μετάβασης του POMDP $P(u)$, με βάση τον οποίο καθορίζεται το x_{k+1} και y_{k+1}, ειναι ο τετραγωνικός πίνακας με στοιχεία $P_{ij}(u) = \mathbb{P}(x_{k+1} = j | x_k = i, u_k = u)$, $i, j \in \mathcal{X}$,

- για κάθε u $\in \mathcal{U}$, το $O_{iy}(u)$ υποδηλώνει την κατανομή που ακολουθούν οι παρατηρήσεις κάθε χρονική στιγμή k, με $O_{iy}(u) = \mathbb{P}(y_{k+1} = y|x_{k+1} = i, u_k = u), i \in \mathcal{X}, y \in \mathcal{Y}$,
- Για κάθε x_k και u_k, ο *decision-maker* επιβαρύνεται με ένα κόστος $c(x_k, u_k)$,
- Την τελευταία χρονική στιγμή του προκαθορισμένου ορίζοντα, $k = N$, ο *decision-maker* επιβαρύνεται με κόστος $c_N(x_N)$.

Δοθέντος του μοντέλου ενός POMDP όπως μόλις περιγράφηκε, ο *decision-maker* επιλέγει κάθε χρονική στιγμή k μια ενέργεια u_k συσσωρεύοντας ένα στιγμιαίο κόστος $c(x_k, u_k)$. Για την επιλογή της καταλληλότερης ενέργειας, ο *decision-maker* χρησιμοποιεί όλη την διαθέσιμη πληροφορία που διαθέτει μέχρι την χρονική στιγμή k, $H_k = \{b_0, u_0, y_1, ..., u_{k-1}, y_k\}$ χρησιμοποιώντας την τρέχουσα πολιτική π_k. Το κριτήριο απόδοσης των αποφάσεων του *decision-maker* κατά το πέρας του προκαθορισμένου χρονικού ορίζοντα είναι

$$J_{\boldsymbol{\pi}}(b_0) = \mathbb{E}_{\boldsymbol{\pi}} \left\{ \sum_{k=0}^{N-1} c(x_k, u_k) + c_N(x_N) | b_0 \right\} \tag{2.1}$$

το οποίο είναι το αναμενόμενο συσσωρευτικό κόστος με το οποίο επιβαρύνεται ο *decision-maker*, δοθείσης μιας αρχικές κατανομής b_0 της αλυσίδας Markov. Στόχος είναι η εύρεση της βέλτιστης σειράς πολιτικών

$$\boldsymbol{\pi}^* = \arg\min_{\boldsymbol{\pi}} J_{\boldsymbol{\pi}}(b_0), \forall b_0 \in \mathcal{B} \tag{2.2}$$

όπου $\mathcal{B}$ ο χώρος[1] που συνθέτουν όλα τα δυνατά beliefs.

2.1.3 Κατάσταση belief & Δυναμικός Προγραμματισμός

Σε αυτήν την υποενότητα ασχολούμαστε πιο διεξοδικά με την έννοια της κατάστασης belief, η οποία αποτελεί την posterior κατανομή των καταστάσεων, όπως προκύπτει από το HMM Φίλτρο (βλ. *Παράτημα Ι*). Έπειτα η βέλτιστη πολιτική διατυπώνεται ως λύση της αντίστοιχης αναδρομικής εξίσωσης Bellman δυναμικού προγραμματισμού, όπου η τελευταία διατυπώνεται ως προς τις καταστάσεις belief, πλέον.

Έχει γίνει γνωστό από το προηγούμενο κεφάλαιο ότι για ένα (πλήρως παρατηρούμενο) MDP, η βέλτιστη πολιτική είναι μαρκοβιανή και η βέλτιστη ενέργεια είναι $u_k = \pi_k^*(H_k)$ με $H_k = \{x_0, u_0, ..., x_{k-1}, u_{k-1}, x_k\}$. Αντιθέτως, στην περίπτωση των POMDPs το ιστορικό μέχρι την χρονική στιγμή k, είναι $H_k = \{b_0, u_0, y_1, ..., u_{k-1}, y_k\}$. Δεδομένου ότι η διάσταση του H_k αυξάνεται με την πάροδο του χρόνου, είναι αναγκαία η εύρεση κάποιας επαρκούς εκτιμήτριας, η διάσταση της οποίας να παραμένει σταθερή. Αποδεικνύεται ότι η posterior κατανομή των καταστάσεων, b_k, η οποία υπολογίζεται μέσω του HMM Φίλτρου, αποτελεί επαρκή στατιστική της ιστορίας H_k. Ακολούθως ορίζεται η posterion κατανομή της υφιστάμενης αλυσίδας Markov, δοθέντος του ιστορικού H_k, ως

$$b_k(i) = \mathbb{P}(x_k = i|H_k), i \in \mathcal{X} \text{ και } H_k = \{b_0, u_0, y_1, ..., u_{k-1}, y_k\} \tag{2.3}$$

Το $b_k = \{b_k(1), ..., b_k(X)\}$ ορίζεται ως η κατάσταση belief και υπολογίζεται με την βοήθεια του HMM Φίλτρου (βλ. *Παράρτημα Ι*) ως εξής

$$b_k = T(b_{k-1}, u_{k-1}, y_k) = \mathbb{P}(x_k|b_{k-1}, u_{k-1}, y_k) \tag{2.4}$$

[1] Χώρος Simplex (βλ. 2.1.3)

$$\text{με } T(b_{k-1}, u_{k-1}, y_k) = \frac{\mathbb{P}(y_k|x_k, u_{k-1}) \left\{ \sum_{x_{k-1} \in \mathcal{X}} \mathbb{P}(x_k|u_{k-1}, x_{k-1}) \mathbb{P}(x_{k-1}|b_{k-1}) \right\}}{\sum_{\tilde{x}_k \in \mathcal{X}} \mathbb{P}(y_k|\tilde{x}_k, u_{k-1}) \left\{ \sum_{x_{k-1} \in \mathcal{X}} \mathbb{P}(\tilde{x}_k|u_{k-1}, x_{k-1}) \mathbb{P}(x_{k-1}|b_{k-1}) \right\}} \tag{2.5}$$

όπου $O_{x_k,y_k}(u_{k-1}) = \mathbb{P}(y_k|x_k, u_{k-1})$, $P_{x_{k-1},x_k}(u_{k-1}) = \mathbb{P}(x_k|u_{k-1}, x_{k-1})$, $b_{k-1} = \mathbb{P}(x_{k-1}|H_{k-1})$.

Είναι φανερό επομένως, ότι η εύρεση της βέλτιστης πολιτικής, π_k^*, μπορεί να γίνει με βάση την κατάσταση belief b_k, έναντι του ιστορικού H_k. Τα $b_k, k = 0, 1, ...$ πρόκειται ουσιαστικά για $|\mathcal{X}|$-διάστατα διανύσματα πιθανότητας. Κατα συνέπεια "ζούν" στον $(|\mathcal{X}| - 1)$-διάστατο μοναδιαίο Simplex,

$$\mathcal{B} = \{b \in \mathbb{R}^{|\mathcal{X}|} : \sum_{i \in \mathcal{X}} b(i) = 1, 0 \leq b(i) \leq 1, \forall i \in \mathcal{X}\} \tag{2.6}$$

ο οποίος καλείται χώρος καταστάσεων (belief space).

Με βάση την *Σχήμα 2.1*, γίνεται πλέον κατανοητό, ότι ένα POMDP αποτελείται από:

- ένα HMM Φίλτρο, το οποίο χρησιμοποιεί θορυβώδεις παρατηρήσεις y_k, προκειμένου να προσδιορίσει την τρέχουσα κατάσταση belief, b_k,
- τον *decision-maker*, ο οποίος πραγματοποιεί το mapping $\boldsymbol{\pi} : \mathcal{B} \to \mathcal{U}$.

Δεδομένου ότι η βέλτιστη πολιτική π_k^* βασίζεται πλέον αποκλειστικά στο b_k, κάθε χρονική στιγμή k, είναι φανερό ότι ο καθορισμός της ολικής βέλτιστης πολιτικής $\boldsymbol{\pi} = \{\pi_0, ..., \pi_{N-1}\}$[2], είναι ισοδύναμος με τον διαμερισμό του χώρου $\mathcal{B}$ σε περιοχές, στις οποίες μια συγκεκριμένη ενέργεια $u \in \mathcal{U}$, είναι βέλτιστη.

Ακολούθως (**Αλγόριθμος 4**) παρουσιάζονται συνοπτικά τα βήματα που ακολουθεί ένας τυπικός POMDP *decision-maker*, κάθε χρονική στιγμή k.

Algorithm 4 POMDP decision-maker

Επιλέξτε $x_0 \sim b_0$
for $k = 0$ to $N - 1$ **do**
  Βήμα 1: Με βάση την κατάσταση belief b_k, επιλέγεται μια ενέργεια $u_k = \pi_k(b_k) \in \mathcal{U}$,
  Βήμα 2: Ο decision-maker επιβαρύνεται με ένα κόστος $b_k^T r_{u_k}$, όπου
  $r_{u_k} = (r(1, u_k), ..., r(|\mathcal{X}|, u_k))^T$,
  Βήμα 3: Η επόμενη κατάσταση επιλέγεται τυχαία, με πιθανότητα μετάβασης
  $P_{x_k,x_{k+1}}(u_k)$,
  Βήμα 4: Ο decision-maker λαμβάνει κάποια θορυβώδη παρατήρηση $y_{k+1} \in \mathcal{Y}$,
  από το περιβάλλον του, με βάση την κατανομή $O_{x_{k+1},y_{k+1}}(u_k)$,
  Βήμα 5: Γίνεται ανανέωση της κατάστασης belief με την βοήθεια του υφιστάμενο
  HMM Φίλτρου, $b_{k+1} = T(b_k, u_k, y_{k+1})$.
end for

Είναι χρήσιμο σε αυτό το σημείο, να εισαχθεί ορισμένος πρόσθετος συμβολισμός, ο οποίος θα μας φανεί ιδιαίτερα χρήσιμος κατά την συζήτηση των *exact* αλγορίθμων. Ειδικότερα, ορίζεται ως

$$\sigma(b_{k-1}, u_{k-1}, y_k) = \mathbb{P}(y_k|b_{k-1}, u_{k-1}) = \sum_{\tilde{x}_k \in \mathcal{X}} \mathbb{P}(y_k|\tilde{x}_k, u_{k-1}) \sum_{x_{k-1} \in \mathcal{X}} \mathbb{P}(\tilde{x}_k|u_{k-1}, x_{k-1}) \mathbb{P}(x_{k-1}|b_{k-1}) \tag{2.7}$$

[2]θεωρούμε POMDP πεπερασμένου ορίζοντα στο σημείο αυτό.

Επίσης ορίζονται οι πιθανότητες μετάβασης των καταστάσεων beliefs, ως εξής

$$\xi(b_{k-1}, u_{k-1}, b_k) = \sum_{y_k \in \mathcal{Y}} \delta_{b_k, T(b_{k-1}, u_{k-1}, y_k)} \sigma(b_{k-1}, u_{k-1}, y_k) \tag{2.8}$$

με

$$\delta_{b_k, T(b_{k-1}, u_{k-1}, y_k)} = \begin{cases} 1 & \textit{αν } b_k = T(b_{k-1}, u_{k-1}, y_k) \\ 0 & \textit{διαφορετικά} \end{cases} \tag{2.9}$$

Λόγω του γεγονότος ότι το POMDP μπορεί να αντιμετωπιστεί ως συνεχούς χώρου καταστάσεων MDP [14, σελ. 151-152], αναμένουμε ότι είναι εφικτή η προσαρμογή των εξισώσεων δυναμικού προγραμματισμού των MDPs, στην κατάλληλη μορφή, ώστε να αντιπροσωπεύουν επαρκώς τα POMDPs.

Δοθείσας μίας κατάστασης belief και υποθέτοντας ότι ο χώρος καταστάσεων και ενεργείων είναι πεπερασμένος, καθίσταται αντιληπτό ότι υπάρχει μόνο ένας πεπερασμένος αριθμός από μελλοντικά beliefs. Ας θεωρηθεί το σύνολο των πιθανών επισκέψιμων beliefs, $\mathcal{B}'(b_{k-1}, u_{k-1}) = \{T(b_{k-1}, u_{k-1}, y_k) | y_k \in \mathcal{Y}\}$. Επίσης, το κόστος με το οποίο ζημειώνεται ο *decision-maker* μπορεί να εκφραστεί ως

$$c(b_k, u_k) = \sum_{x_k \in \mathcal{X}} b_k(x_k) r(x_k, u_k) = r_{u_k}^T b_k \tag{2.10}$$

Χρησιμοποιώντας την (1.7) με τις κατάλληλες αντικαταστάσεις οδηγούμαστε:

$$\begin{aligned} V_n(i) &= \min_{u \in \mathcal{U}} \left\{ c(b,u) + \sum_{b' \in \mathcal{B}'(b,u)} \xi(b,u,b') V_{n-1}(b') \right\} \\ &= \min_{u \in \mathcal{U}} \left\{ c(b,u) + \sum_{y \in \mathcal{Y}} \sigma(b,u,y) V_{n-1}(T(b,u,y)) \right\} \\ &= \min_{u \in \mathcal{U}} \left\{ r_u^T b + \sum_{y \in \mathcal{Y}} \sigma(b,u,y) V_{n-1}(T(b,u,y)) \right\} \end{aligned} \tag{2.11}$$

Συνεπώς εκμεταλλευόμενοι το γεγονός ότι ένα POMDP μπορεί να θεωρηθεί ως MDP συνεχούς χώρου καταστάσεων, εκφράσαμε την αντίστοιχη της (1.7) ως (2.11). Το ακόλουθο θεώρημα αποτελεί το ανάλογο του θεωρήματος 1.1

Θεώρημα 2.1: Θεωρώντας ένα POMDP πεπερασμένου χρονικού ορίζοντα, τότε

- Το ελάχιστο συσσωρευτικό κόστος, $J_{\boldsymbol{\pi}^*}(b)$, με το οποίο ζημιώνεται ο *decision-maker* μπορεί να επιτευχθεί μέσω ντετερμινιστικών πολιτικών

$$\boldsymbol{\pi}^* = \{\pi_0, ..., \pi_{N-1}\}, \text{ με } u_k = \pi_k^*(b_k) \tag{2.12}$$

- Η βέλτιστη πολιτική $\boldsymbol{\pi}^* = \{\pi_0, ..., \pi_{N-1}\}$ είναι λύση της ακόλουθης Bellman αναδρομής δυναμικού προγραμματισμού[3]

$$V_0(b) = r_N^T b \text{ και έπειτα για } n = 1, ..., N-1 \tag{2.13}$$

[3]θα μπορούσε, σε αντιστοιχία με το *Θεώρημα 1.1*, να δοθεί η κλασσική προς τα πίσω αναδρομή Bellman. Προτιμήθηκε η προς τα εμπρός αναδρομή, προκειμένου να διευκολυνθεί η συζήτηση μας, στην επόμενη υποενότητα, σχετικά με τους *exact* αλγορίθμους.

$$V_n(b) = \min_{u \in \mathcal{U}} V_n(b,u), \text{ με } V_n(b,u) = r_u^T b + \sum_{y \in \mathcal{Y}} \sigma(b,u,y) V_{n-1}(T(b,u,y)), \tag{2.14}$$

$$\pi_n^*(b) = \arg\min_{u \in \mathcal{U}} \left\{ r_u^T b + \sum_{y \in \mathcal{Y}} \sigma(b,u,y) V_{n-1}(T(b,u,y)) \right\} \tag{2.15}$$

Ακόμη, σχετικά με την παρουσίαση των *exact* αλγορίθμων, είναι αρκετά βολικό να εκφράσουμε τις (2.13) - (2.15) ως εξής:

$$V_n(b) = \min_{u \in \mathcal{U}} V_n(b,u), \tag{2.16}$$

$$V_n(b,u) = \sum_{y \in \mathcal{Y}} V_n(b,u,y), \tag{2.17}$$

$$V_n(b,u,y) = \frac{c(b,u)}{|\mathcal{Y}|} + \sigma(b,u,y) V_{n-1}(T(b,u,y)) \tag{2.18}$$

Το κύριο ερώτημα πλέον είναι, πώς μπορεί να υπολογιστεί η βέλτιστη πολιτική για το POMDP πεπερασμένου ορίζοντα, δεδομένου ότι ο χώρος καταστάσεων είναι ο μοναδιαίος Simplex $\mathcal{B}$. Όπως έχει αναφερθεί, η εύρεση της βέλτιστης πολιτικής είναι ισοδύναμη με τον διαχωρισμό του χώρου των beliefs, σε περιοχές. Έχει δειχθεί [23, 26], ότι η βέλτιστη value function πεπερασμένου ορίζοντα, είναι τμηματικά γραμμική και κυρτή/κοίλη (piecewise liner and convex/concave - PWLC), για οποιονδήποτε χρονικό ορίζοντα T. Αυτή η piecewise linear ιδιότητα είναι ιδιαίτερα χρήσιμη, καθώς επιτρέπει την ακριβή αναπαράσταση της value function, στο πλαίσιο του POMDP πεπερασμένου ορίζοντα.

Κάθε γραμμικό τμήμα της PWLC value function που περιγράφεται, είναι ένα υπερεπίπεδο στον $|\mathcal{X}|-$διάστατο χώρο και μπορεί να αναπαρασταθεί με ένα $|\mathcal{X}|-$διάστατο διάνυσμα συντελεστών. Ένα γραμμικό τμήμα της PWLC value function θα αναπαριστάται ως γ, ενώ το i-οστό στοιχείο του ως $\gamma(i)$ και με Γ αναπαρίσταται το σύνολο των διανυσμάτων που συνθέτουν την PWLC value function V.

Η κυρτότητα (ή κοιλότητα)[4], υποδηλώνει ότι η value function αποτελεί την ανώτερη (ή κατώτερη) επιφάνεια από τα υφιστάμενα διανύσματα. Θεωρώντας πρόβλημα ελαχιστοποίησης, η τιμή της value function σε μια κατάσταση belief μπορεί να υπολογιστεί ως

$$V(b) = \min_{\gamma \in \Gamma} \left\{ \sum_{i \in \mathcal{X}} b(i)\gamma(i) \right\} = \min_{\gamma \in \Gamma} \gamma^T b \tag{2.19}$$

Το ακόλουθο θεώρημα δείχνει ότι η εξίσωση Bellman (2.14), για POMDP πεπερασμένου ορίζοντα, έχει ένα πεπερασμένης διάστασης χαρακτηρισμό, όταν ο χώρος των παρατηρήσεων είναι πεπερασμένος. Ειδικότερα

Θεώρημα 2.2: Έστω ένα POMDP με πεπερασμένο χώρο καταστάσεων, ενεργειών και παρατηρήσεων. Τότε για την value function $V_n(b)$ της εξίσωσης Bellman (2.14) και την αντίστοιχη βέλτιστη πολιτική $\pi_n^*(b)$ (2.15), κάθε χρονική στιγμή n, ισχύει:

- Η $V_n(b)$ είναι piecewise linear concave (ή convex) ως προς $b \in \mathcal{B}$. Συνεπώς

$$V_n(b) = \min_{\gamma \in \Gamma_n} \gamma^T b \tag{2.20}$$

όπου Γ_n αντιπροσωπεύει το σύνολο των διαθέσιμων διανυσμάτων που χαρακτηρίζουν την value function, την χρονική στιγμή n.

[4] ανάλογα με το εάν το πρόβλημα είναι μεγιστοποίησης ή ελαχιστοποίησης, αντίστοιχα.

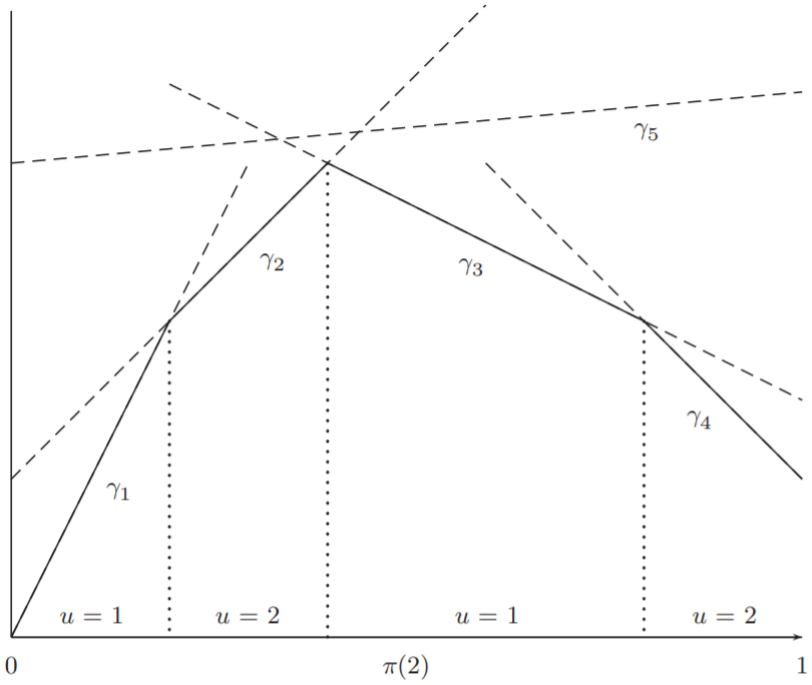
- Ο χώρος καταστάσεων belief $\mathcal{B}$ μπορεί να διαμεριστεί το πολύ σε $|\Gamma_n|$ κυρτά πολύτοπα. Σε κάθε ένα από αυτά τα πολύτοπα $\mathcal{R}_l = \{b : V_n(b) = \gamma_l^T b\}$, η βέλτιστη πολιτική $\pi_n^*(b)$ είναι σταθερή και αντιστοιχεί σε μία συγκεκριμένη ενέργεια $u_n^l \in \mathcal{U}$. Δηλαδή, για κάθε $b \in R_l$ η βέλτιστη πολιτική είναι

$$\pi_n^*(b) = u(\arg\min_{\gamma_l \in \Gamma_n} \gamma_l^T b) \tag{2.21}$$

όπου το δεξιό μέρος της (2.21) αντιπροσωπεύει την ενέργεια που σχετίζεται με την συγκεκριμένη περιοχή του χώρου Simplex $\mathcal{B}$, $\mathcal{R}_l$

Απόδειξη: [14, σελ. 155-156]

Ακολούθως παρουσιάζεται σχηματικά η value function ενός POMDP με $|\mathcal{X}| = 2$ και $|\mathcal{U}| = 2$



Σχήμα 2.2: Piecewise linear concave value function προβλήματος ελαχιστοποίησης, με $|\mathcal{X}| = 2$ και $|\mathcal{U}| = 2$

Όσον αφορά την *VI μέθοδο* για POMDPs, η ιδέα θεωρητικά είναι ίδια με τα MDPs. Παρ' όλα αυτά, ο συνεχής χώρος καταστάσεων απαγορεύει την πραγματοποίηση επαναλήψεων πάνω στις καταστάσεις belief. Η κύρια δυσκολία στην εφαρμογή αυτής καθ' αυτής της ιδέας, έγκειται στον υπολογισμό των V_n, V_{n-1} στην (2.18). Στην υποενότητα που ακολουθεί, θα συζητηθούν αλγόριθμοι που υπερβαίνουν αυτή την δυσκολία και πραγματοποιούν τον 1-βήματος δυναμικό προγραμματισμό[5], προκειμένου να εντοπίσουν την βέλτιστη V^*.

Όσον αφορά την *PI μέθοδο*, δεν υπάρχει εγγύηση ύπαρξης της ακριβούς μεθόδου για οποιοδήποτε POMDP πεπερασμένου χρονικού ορίζοντα. Συνεπώς, η υιοθέτηση προσεγγιστικών τεχνικών είναι αναγκαία.

[5]Πέραν του γεγονότος ότι αποτελεί την βάση της *VI μεθόδου*, επιπλέον χρησιμοποιείται σε αρκετά προσεγγιστικά σχήματα, τα οποία δεν βασίζονται τόσο άμεσα με την *VI μέθοδο*.

2.1.4 POMDP discounted κόστους με άπειρο ορίζοντα

Μέχρι στιγμής έγινε αναφορά αποκλειστικά σε POMDPs πεπερασμένου χρονικού ορίζοντα. Αντίθετα, σε αυτή την υποενότητα θα συζητηθεί το *discounted* POMDPs άπειρου ορίζοντα, το οποίο αποτελεί επέκταση των MDPs πεπερασμένου ορίζοντα (βλ. 1.2.4). Όπως είναι αναμενόμενο, δεν είναι δυνατό πλέον να θεωρούνται οι πιθανότητες μετάβασης καθώς και το κόστος με το οποίο ζημειώνεται ο *decision-maker*, τόσο ξεκάθαρα εξαρτώμενα από τον χρόνο, ενώ ταυτόχρονα, προφανώς, δεν υφίσταται τερματικό κόστος.

Παρόμοια με τα *Discounted MDPs*, θα μας απασχολήσουν αποκλειστικά *στάσιμες πολιτικές*. Δοθείσας κάποιας αρχικής κατάστασης belief $b_0 \in \mathcal{B}$ και του *discounted* παράγοντα, ρ, η αντικειμενική συνάρτηση ορίζεται ως

$$J_\pi(b_0) = \mathbb{E}_\pi \left\{ \sum_{k=0}^{\infty} \rho^k c(x_k, \pi(b_k)) \right\} = \mathbb{E}_\pi \left\{ \sum_{k=0}^{\infty} \rho^k r_{\pi(b_k)}^T b_k \right\} \tag{2.22}$$

με $J_\pi(b_0)$ να αντιπροσωπεύει το discounted αναμενόμενο κόστος που θα επιβαρυνθεί ο *decision-maker* σε άπειρο χρονικό ορίζοντα, ξεκινώντας από το belief b_0.

Στόχος είναι η εύρεση της βέλτιστης $\pi^* : \mathcal{B} \to \mathcal{U}$ προκειμένου $J_{\pi^*}(b_0) \leq J_\pi(b_0), \forall b_0 \in \mathcal{B}$. Το ακόλουθο θεώρημα αποτελεί το αντίστοιχο του *Θεωρήματος 1.2* (βλ. 1.2.2)

Θεώρημα 2.3: Έστω ένα POMDP discounted κόστους άπειρου ορίζοντα, με discounted παράγοντα $\rho \in [0, 1)$. Τότε

- Το βέλτιστο συσσωρευτικό κόστος με το οποίο ζημειώνεται ο *decision-maker* επιτυγχάνεται ακολουθώντας κάποια *στάσιμη* ντετερμινιστική μαρκοβιανή πολιτική π^*,
- Η βέλτιστη πολιτική $\pi^*(b)$ και η αντίστοιχη value function $V(b)$, ικανοποιούν

$$V(b) = \min_{u \in \mathcal{U}} Q(b, u), \; Q(b, u) = r_u^T b + \rho \sum_{y \in \mathcal{Y}} \sigma(b, u, y) V(T(b, u, y)) \tag{2.23}$$

$$\pi^*(b) = \arg\min_{u \in \mathcal{U}} Q(b, u), \; J_{\pi^*}(b_0) = V(b_0), \tag{2.24}$$

- Η value function είναι συνεχής κοίλη στο $\mathcal{B}$.

Απόδειξη: [14, σελ. 165]

Επιπλέον, ας θεωρηθεί οποιαδήποτε φραγμένη συνάρτηση $\tilde{f} \in \mathcal{B}$ και ορίζεται ο τελεστής δυναμικού προγραμματισμού $D : \; \tilde{f} \to \mathbb{R}$ ως

$$D\tilde{f}(b) = \min_{u \in \mathcal{U}} \left\{ r_u^T b + \rho \sum_{y \in \mathcal{Y}} \sigma(p, u, y) \tilde{f}(T(b, u, y) \right\} \tag{2.25}$$

Με βάση το ακόλουθο θεώρημα παρουσιάζεται ένα βασικό αποτέλεσμα σχετικά με την ύπαρξη και την μοναδικότητα της λύσης V^* που ικανοποιεί τις (2.23) - (2.24).

Θεώρημα 2.4: Για discounted παράγοντα $\rho \in [0, 1)$ και για οποιεσδήποτε $\varphi, \zeta \in \mathcal{B}(\mathcal{X})$, με $\mathcal{B}(\mathcal{X})$: το σύνολο των πραγματικών φραγμένων συναρτήσεων επάνω από τον χώρο των beliefs $\mathcal{B}$, ισχύει

$$\|D\varphi - D\zeta\|_\infty \leq \rho \|\varphi - \zeta\|_\infty \tag{2.26}$$

πράγμα που υποδηλώνει ότι ο D αποτελεί ένα *contraction mapping*.

Απόδειξη: [14, 165-166]

Ως άμεσο επακόλουθο από το *θεώρημα σταθερού σημείου Banach*, υπάρχει πάντοτε μια λύση V που ικανοποιεί την εξίσωση Bellman (2.23), $V = DV$.

Όσον αφορά την *VI μέθοδο*, μπορεί θεωρητικά να εφαρμοστεί προκειμένου να προσεγγιστεί η $V(b)$, έπειτα από ένα πλήθος επαναλήψεων N. Όπως όμως έχει προαναφερθεί, δοθέντος του γεγονότος ότι το πλήθος των τμηματικά γραμμικών τμημάτων αυξάνει εκθετικά στην εκάστοτε επανάληψη, καθίσταται σαφές πως στην πράξη δεν είναι αποδεκτή η χρήση της *VI* μεθόδου, αλλά απαιτείται να χρησιμοποιηθούν προσεγγιστικοί αλγόριθμοι, με στόχο την επίλυση της εξίσωσης Bellman (2.23).

Από την άλλη πλευρά, χρησιμοποιώντας την *PI μέθοδο* δεν είναι πάντα δυνατό να προσδιοριστεί η V^*. Στην περίπτωση των *στάσιμων finitely transient* πολιτικών όμως, υπάρχει εγγύση ότι το $J_\pi(b) = \lim_{n\to\infty} V_{\pi,n}(b)$ είναι PWLC με πεπερασμένο αριθμό τμηματικά γραμμικών τμημάτων στον $\mathcal{B}$ και μπορεί να υπολογιστεί επακριβώς επιλύοντας ένα σύστημα γραμμικών εξισώσεων. Ως συνέπεια είναι δυνατόν το $J_\pi(b)$ να υπολογιστεί επακριβώς, χρησιμοποιώντας την *PI μέθοδο*. Η περαιτέρω ανάλυση, όμως, υπερβαίνει τον σκοπό της συγκεκριμένης εργασίας και ο αναγνώστης παραπέμπεται ενδεικτικά [6, 327-341; 14, 168-169]

2.2 Κατηγορίες Αλγορίθμων Επίλυσης POMDPs

Στην συγκεκριμένη ενότητα, θα συζητηθούν συνοπτικά οι βασικές κατηγορίες αλγορίθμων που επιλύουν POMDPs. Αρχικά δίνεται ιδιαίτερη βαρύτητα στους *exact* αλγορίθμους, οι οποίοι επιλύουν βέλτιστα POMDPs πεπερασμένου χρονικού ορίζοντα. Έπειτα, λαμβάνοντας υπόψη το έντονο υπολογιστικό κόστος που συνδέεται άρρηκτα με τους *exact* αλγορίθμους, προβάλλει επιτακτική η συζήτηση προσεγγιστικών αλγορίθμων, οι οποίοι επιλύουν ικανοποιητικά POMDPs πολύ μεγαλύτερου μεγέθους. Θα συζητηθούν κυρίως δύο βασικές οικογένειες προσεγγιστικών αλγορίθμων, οι *Point-Based (PB)* και οι *online* αλγόριθμοι.

2.2.1 Exact Αλγόριθμοι

Οι *exact* αλγόριθμοι επιλύουν βέλτιστα POMDPs πεπερασμένου χρονικού ορίζοντα, βασιζόμενα στους ισχυρισμούς που παρουσιάζονται στο *Θεώρημα 2.2*. Η ανάπτυξη τους βασίστηκε στην ιδέα του Sondik [26], όπου και αποτέλεσε τον πρώτο *exact* αλγόριθμο για την επίλυση POMDPs πεπερασμένου ορίζοντα.

Ειδικότερα, βασιζόμενοι στις (2.16) - (2.18), το σύνολο των διανυσμάτων Γ_n που συνθέτουν την value function την χρονική στιγμή n μπορεί να κατασκευαστεί ως εξής

$$\Gamma_n(u,y) = \left\{ \frac{r_u}{|\mathcal{Y}|} + P(u)O_y(u)\gamma | \gamma \in \Gamma^{n-1} \right\} \tag{2.27}$$

$$\Gamma_n(y) = \oplus_y \Gamma_n(u,y) \tag{2.28}$$

$$\Gamma_n = \cup_{u\in\mathcal{U}} \Gamma_n(u) \tag{2.29}$$

όπου, εάν θεωρηθούν δύο σέτ από διανύσματα, έστω A και B, τότε $A \oplus B = \{a + b \mid a \in A, b \in B\}$.

Είναι φανερό ότι το σέτ Γ_n μπορεί να περιέχει διανύσματα που είναι "μη ενεργά", με την έννοια ότι σε καμία περιοχή του χώρου των beliefs, δεν συνεισφέρουν στην value function V_n. Με βάση την ιδέα του Sodnik, σε συνδυασμό με την εξάλειψη μη χρήσιμων διανυσμάτων, παρουσιάζεται στην συνέχεια ο πιο δημοφιλής και αποδοτικός *exact* αλγόριθμος, ο *Incremental Pruning*.

Στόχος, λοιπόν, είναι να αναπαρασταθεί η value function με μία πιο συμπαγή μορφή, μέσω του pruning των "μη ενεργών" διανυσμάτων. Έστω λοιπόν $\Gamma_n^{pruned} = prune(\Gamma_n)$. Αποδεικνύεται [6] ότι ισχύει

$$\begin{aligned} \Gamma_n^{pruned}(u) = prune(\Gamma_n(u, y_1) \oplus ... \oplus \Gamma_n(u, y_{|\mathcal{Y}|})) \\ = prune(prune(\Gamma_n(u, y_1) \oplus \Gamma_n(u, y_2))... \oplus \Gamma_n(u, y_{|\mathcal{Y}|})) \end{aligned} \tag{2.30}$$

Βασιζόμενοι σε αυτήν την παρατήρηση και στις (2.27-2.29), παρουσιάζεται ακολούθως (**Αλγόριθμος 5**), σε μορφή ψευδογλώσσας, ο αλγόριθμος *Incremental Pruning*.

Algorithm 5 Incremental Pruning (2.27 - 2.29)

Δίνεται ένα σέτ διανυσμάτων Γ_n
$\Gamma_{n+1}(u, y) \leftarrow \{\emptyset\}, \Gamma_{n+1}(u) \leftarrow \{\emptyset\}, \Gamma_{n+1} \leftarrow \{\emptyset\}$
for each $u \in \mathcal{U}$ **do**
  for each $y \in \mathcal{Y}$ **do**
    $\Gamma_{n+1}(u, y) \leftarrow prune(\left\{ \frac{r_u}{|\mathcal{Y}|} + P(u)O_y(u)\gamma | \gamma \in \Gamma^n \right\})$
    $\Gamma_{n+1}(u) \leftarrow prune(\Gamma_{n+1}(u) \oplus \Gamma_{n+1}(u, y))$
  end for
end for
$\Gamma_{n+1} \leftarrow prune(\Gamma_{n+1} \cup \Gamma_{n+1}(u))$
return Γ_{n+1}

Η συνάρτηση $prune$ πραγματοποιείται σε δύο φάσεις. Αρχικά απομακρύνονται τα διανύσματα που "καλύπτονται" πλήρως[6] από κάποιο άλλο διάνυσμα στο σέτ Γ_n. Έπειτα στόχος είναι να εντοπιστούν και να απομακρυνθούν τα διανύσματα που δεν "καλύπτονται" από ένα μόνο διάνυσμα, αλλά από περισσότερα. Για τον σκοπό αυτό χρησιμοποιείται το ακολουθό πρόβλημα γραμμικού προγραμματισμού

$$min\ x \tag{2.31}$$

$$\text{s.t. } (\gamma - \tilde{\gamma})^T b \geq x, \forall \tilde{\gamma} \in \Gamma_n - \{\gamma\}, b \in \mathcal{B} \tag{2.32}$$

Συνεπώς, εάν το προαναφερθέν πρόβλημα γραμμικού πργραμματισμού καταλήξει σε ≥ 0, τότε το γ "καλύπτεται" από άλλα διανύσματα του Γ_n και συνεπώς δεν είναι χρήσιμο. Το pruning κάνει τον *Incremental Pruning* αποδοτικότερο σε σχέση με άλλους *exact* αλγορίθμους, καθώς με τον τρόπο αυτό περιορίζεται ο πλήθος μη χρήσιμων εν τέλει διανυσμάτων, μειώνοντας το αρχικά υφιστάμενο υπολογιστικό κόστος.

Τέλος, υπάρχουν και άλλοι ακόμα αλγόριθμοι οι οποίοι απαρτίζουν την οικογένεια των *exact* αλγορίθμων, όπως ο *αλγόριθμος του Monahan*, ο *Witness αλγόριθμος* κ.α. [14, 6].

[6] Θεωρώντας τα δύο διανύσματα a, a' $\in \Gamma_n$ και έστω ότι το a επικαλύπτεται πλήρως από το a'. Τότε θα ισχύει $a(i) \leq a'(i), \forall i \in \mathcal{X}$.

2.2.2 Point-Based Value Interation Αλγόριθμοι

Οι *Point-Based* αλγόριθμοι επιχειρούν να υπολογίσουν μία προσέγγιση της βέλτιστης value function, σε συγκεκριμένα σημεία του χώρου καταστάσεων $\mathcal{B}$. Ειδικότερα, επικεντρώνονται σε ένα σέτ από beliefs που έχουν επισκεφθεί μέχρι κάποια χρονική στιγμή, επιλύοντας το POMDP πρόβλημα αποκλειστικά σε εκείνα.

Αναλογιζόμενοι το γεγονός ότι το πλήθος του συνόλου των διανυσμάτων που αντιπροσωπεύουν την value function, αυξάνεται εκθετικά ανά επανάληψη, προβάλλει επιτακτική η ανάγκη περιορισμού του. Προφανώς όμως, υπάρχει ένα *trade-off* μεταξύ του υφιστάμενου υπολογιστικού κόστους και της επιθυμητής ακριβείας προσέγγισης της value function που απαιτείται.

Η βασική ιδέα των *Point-Based* αλγορίθμων είναι ο υπολογισμός της τιμής της βέλτιστης value function V σε κάποιο συγκεκριμένο belief b, η οποία έπεται της εκτέλεσης ενός βήματος δυναμικού προγραμματισμού Bellman (2.23), με βάση μία δοσμένη value function. Ακολούθως παρουσιάζεται ο τρόπος με τον οποίο μπορεί να πραγματοποιηθεί αποδοτικά η εύρεση αυτής της τιμή, σε συνδυασμό με το βέλτιστο $\gamma-$διάνυσμα που αντιπροσωπεύει την V στο b

$$V_{new}(b) = \min_{u\in\mathcal{U}} \left\{ r_u^T b + \rho \sum_{y\in\mathcal{Y}} \sigma(b,u,y) V_{prev}(T(b,u,y)) \right\} \tag{2.33}$$

$$= \min_{u\in\mathcal{U}} \left\{ r_u^T b + \rho \sum_{y\in\mathcal{Y}} \sigma(b,u,y) \min_{\gamma\in\Gamma_{prev}} \gamma^T T(b,u,y) \right\} \tag{2.34}$$

$$= \min_{u\in\mathcal{U}} \left\{ r_u^T b + \rho \sum_{y\in\mathcal{Y}} \sigma(b,u,y) \min_{\gamma\in\Gamma_{prev}} \left\{ \frac{\sum_{\tilde{x}\in\mathcal{X}} \gamma(\tilde{x}) O_{\tilde{x},y}(u) \sum_{x\in\mathcal{X}} P_{x,\tilde{x}}(u) b(x)}{\sigma(b,u,y)} \right\} \right\} \tag{2.35}$$

$$= \min_{u\in\mathcal{U}} \left\{ r_u^T b + \rho \sum_{y\in\mathcal{Y}} \min_{\gamma\in\Gamma_{prev}} \left\{ \sum_{x\in\mathcal{X}} b(x) \sum_{\tilde{x}\in\mathcal{X}} \gamma(\tilde{x}) O_{\tilde{x},y}(u) P_{x,\tilde{x}}(u) \right\} \right\} \tag{2.36}$$

$$= \min_{u\in\mathcal{U}} \left\{ r_u^T b + \rho \sum_{y\in\mathcal{Y}} \min_{\gamma\in\Gamma_{prev}} (\gamma^{u,y})^T b \right\} \tag{2.37}$$

με

$$\gamma^{u,y}(x) = \sum_{\tilde{x}\in\mathcal{X}} \gamma(\tilde{x}) O_{\tilde{x},y}(u) P_{x,\tilde{x}}(u), x \in \mathcal{X} \tag{2.38}$$

Είναι ιδιαίτερα βολικό να οριστεί ο τελεστής $backup(B,b)$ ως

$$backup(B,b) = \underset{\gamma^{u,b}:u\in\mathcal{U},\gamma\in\Gamma}{\arg\min}\ b\,\gamma^{u,b} \tag{2.39}$$

με $\gamma^{u,b} = r_u + \rho \sum_{y\in\mathcal{Y}} \left\{ \arg\min_{\gamma^{u,y}:\gamma\in\Gamma} (\gamma^{u,y})^T b \right\}$.

Ο τελεστής $backup(B,b)$ δημιουργεί ένα καινούργιο $\gamma-$διάνυσμα για το εκάστοτε τρέχον b, ενώ απομακρύνει τα διανύσματα που "κυριαρχούνται" από άλλα διανύσματα (prunning), μέσω των δύο τελεστών $\arg\min$.

Η συγκεκριμένη οικογένεια αλγορίθμων, επιχειρεί να προσεγγίσει την value function, υπολογίζοντας την ακριβή τιμή της σε ένα πεπερασμένο σύνολο καταστάσεων beliefs. Ακολούθως παρουσιάζεται η τυπική δομή των *point-based* αλγορίθμων

Algorithm 6 Τυπική Δομή Point-Based Αλγορίθμων

1: **do**
2: **Βήμα 1:** Συλλογή νέων καταστάσεων beliefs και ανανέωση του σέτ B, σε B'
3: **Βήμα 2:** Ανανέωση Γ, χρησιμοποιώντας τον τελεστή $backup(B, b), \forall b \in B'$
4: **while** *Κριτήριο Τερματισμού Ψευδές*

Οι αλγόριθμοι αυτοί απαιτούν να είναι διαθέσιμη μια αρχική εκτίμηση της value function. Υπάρχουν διάφορες προσεγγιστικές μέθοδοι, οι οποίες δύναται να χρησιμοποιηθούν στην πράξη, με την απόδοση τους να εξαρτάται από το αρχικό επιθυμητό υπολογιστικό κόστος που δαπανάται. Για περισσότερο αναλυτική επισκόπηση του θέματος, ένα καλό σημείο αφετηρίας αποτελεί [7].

Υπάρχουν αρκετοί αξιόγογοι *point-based* αλγόριθμοι, όπως για παράδειγμα οι HSVI [24, 25], FSVI [8], GapMin [20], SARSOP [15] κ.α. Στην ακόλουθη ενότητα θα δοθεί ειδική μνεία στον αλγόριθμο *SARSOP*, λόγω της αρκετά καλής κλιμάκωσής του σε προβλήματα μεγαλύτερου μεγέθους, σχέση με τους υπόλοιπους *point-based* αλγορίθμους.

Γενικά μιλώντας, οι *point-based* αλγόριθμοι ανήκουν στην γενικότερη κατηγορία των *offline* αλγορίθμων, καθώς κατασκευάζουν μια πολιτική π, διαχωρίζοντας τον χώρο $\mathcal{B}$ σε περιοχές, ανάλογα με τις σχέσεις κυριαρχίας μεταξύ των διανυσμάτων που συνθέτουν το σύνολο Γ.

2.2.3 Online Αλγόριθμοι

Ως γνωστόν τα POMDPs παρέχουν μια πλούσια δομή για κατ' εξακολούθηση λήψη αποφάσεων υπο αβεβαιότητα, σε στοχαστικά περιβάλλοντα. Λόγω της υφιστάμενης πολυπλοκότητάς τους, η ακριβής επίλυσή (π.χ. *exact* αλγόριθμοι) τους περιορίζονται μόνο σε πολύ μικρά προβλήματα, χωρίς κανένα πρακτικό ενδιαφέρον. Αντίθετα, οι *online* αλγόριθμοι επιτυγχάνουν να μετριάσουν το υφιστάμενο υπολογιστικό κόστος, υπολογίζοντας καλές τοπικές πολιτικές ανά βήμα απόφασης. Κάτι τέτοιο επιτυγχάνεται με μια lookahead αναζήτηση, προκειμένου να εντοπίζεται η καλύτερη δυνατή ενέργεια προς υιοθέτηση, στο εκάστοτε βήμα απόφασης.

Αρκετά συνηθισμένο είναι να χρησιμοποιούνται προσεγγιστικοί *offline* αλγόριθμοι, προκειμένου να υπολογιστούν κάποια άνω και κάτω φράγματα της βέλτιστης value function. Τα φράγματα αυτά χρησιμοποιούνται, έπειτα, ώστε οι *online* αλγόριθμοι να κατευθύνονται προς τις πολλά υποσχόμενες περιοχές του χώρου καταστάσεων belief $\mathcal{B}$, γλιτώντας περιττές αναζητήσεις σε "φτωχές" φαινομενικά περιοχές. Ως επί το πλείστον, προτιμούνται προσεγγιστικές μέθοδοι αρκετά χαμηλού υπολογιστικού κόστους, για τον υπολογισμό των προαναφερθέντων φραγμάτων.

Οι *online* αλγόριθμοι επιχειρώντας την μετρίαση του υπολογιστικού κόστους, προσπαθούν να κατασκευάσουν μια καλή τοπική πολιτική, επικεντρωνόμενοι στα beliefs που είναι επισκέψιμα, από το τρέχον belief. Κατ' αυτόν τον τρόπο, οι απαραίτητοι υπολογισμοί πραγματοποιούνται σε ένα σχετικά μικρού μεγέθους σέτ από beliefs.

Οι αλγόριθμοι που ανήκουν στην συγκεκριμένη οικογένεια, διαθέτουν τυπικά δύο βασικά βήματα: *1) την φάση περιήγησης* (planning phase) και *2) την φάσης εκτέλεσης* (execution phase), τα οποία εκτελούνται κάθε χρονική στιγμή.

Κατά την φάση περιήγησης, δοθέντος του τρέχοντος belief, ο αλγόριθμος υπολογίζει την καλύτερη δυνατή ενέργεια προς εκτέλεση. Κάτι τέτοιο πραγματοποιείται σε δύο κύριες

υποφάσεις. Αρχικά κατασκευάζεται ένα δέντρο, το οποίο αποτελείται από διάφορες πιθανές καταστάσεις beliefs, οι οποίες προκύπτουν έπειτα από την υιοθέτηση μιας σειράς ενεργειών-παρατηρήσεων, έχοντας ως αφετηρία την τρέχουσα κατάσταση belief, η οποία βρίσκεται στην ρίζα του δέντρου. Έπειτα από την επιλογή κάποιου δυνατού ζεύγους ενέργειας-κατάστασης, προστίθεται ως OR-κόμβος στο δέντρο το προκύπτον belief, ενώ μεταξύ των δύο επιπέδων από beliefs, παρεμβάλλεται ως AND-κόμβος η αντίστοιχη ενέργεια που επιλέχθηκε. Έπειτα η τιμή στο τρέχον belief προσεγγίζεται από την διάδοση των τιμών από κάτω προς τα πάνω στο δέντρο, με βάση την εξίσωση Bellman (2.23). Ορισμένες μέθοδοι, επιπλέον, διατηρούν σε κάθε belief κόμβο, ένα άνω και κάτω φράγμα της τιμής της value function, στο συγκεκριμένο belief.

Κατά την φάση εκτέλεσης, η οποία έπεται της φάσης περιήγησης, πραγματοποιείται η επιλογή της καλύτερης δυνατής ενέργειας την συγκεκριμένη χρονική στιγμή και ανανεώνεται η τρέχουσα κατάσταση belief καθώς και το σχηματισμένο δέντρο, ανάλογα με την λαμβανόμενη παρατήρηση.

Ακολούθως, παρουσιάζεται η τυπική δομή των αλγορίθμων που εμπίπτουν στην συγκεκριμένη οικογένεια (**Αλγόριθμος 7**).

Algorithm 7 Τυπική Δομή Online Αλγορίθμων

1: Επιλέξτε κάποια αρχική κατάσταση belief b_0
2: $b_{current} \leftarrow b_0$
3: Το δέντρο $\mathcal{T}$ διαθέτει ως μοναδικό κόμβο-ρίζα το $b_{current}$
4: **do**
5: **do**
6: $b^* \leftarrow ChooseNextNodeToExpand()$
7: $Expand(b^*, D)$
8: $UpdateAncestors(b^*)$
9: **while not** $PlanningIsTerminated()$
10: Εκτελέστε την καλύτερη ενέργεια u^* από τον κόμβο $b_{current}$
11: Λάβετε παρατήρηση o, έπειτα από την εκτέλεση της u^*
12: $b_{current} \leftarrow T(b_{current}, u^*, o)$
13: Ανανεώστε το δέντρο $\mathcal{T}$, έτσι ώστε το νέο $b_{current}$ να αποτελεί την κορύφη του
14: **while not** $ExecutionIsTerminated()$

Η $Expand$ μέθοδος δημιουργεί τα προσβάσιμα beliefs από τον κόμβο b^*, μέχρι κάποιο προκαθορισμένο βάθος περιήγησης D, ενώ ταυτόχρονα εκτιμά την value function, προσεγγιστικά, για τα καινούργια beliefs που δημιουργούνται. Αυτή η προσεγγιστική τιμή της value function διαδίδεται από κάτω προς τα πάνω στο δέντρο $\mathcal{T}$, με την βοήθεια της μεθόδου $UpdateAncestors$. Επιπλέον, κατά την φάση της εκτέλεσης επιλέγεται η καλύτερη ενέργεια που προέκυψε από την φάση περιήγησης, u^*, και σε συνδυασμό με την λαμβανόμενη παρατήρηση, από το περιβάλλον, ανανεώνεται η κατάσταση belief και το αντίστοιχο δέντρο $\mathcal{T}$.

Προκειμένου οι *Online* αλγόριθμοι να αποτελούν μια ανταγωνιστική και αποδοτική επιλογή, δίνεται ιδιαίτερη έμφαση στον περιορισμό του αριθμού των προσβάσιμων beliefs, από το $b_{current}$, στο δέντρο $\mathcal{T}$. Υπάρχουν διάφορες προσεγγίσεις προς αυτή την κατεύθυνση, οι οποίες διαφέρουν στον τρόπο με τον οποίο υλοποιούνται οι μέθοδοι $Expand$ και $ChooseNextNodeToExpand$.

Παρόλο αυτά, ανάλογα με τον τρόπο που πραγματοποιείται η περιήγηση εντός του δέντρου $\mathcal{T}$, οι *online* αλγόριθμοι ομαδοποιούνται σε τρείς βασικές κατηγορίες. Εκείνους που

χρησιμοποιούν:

- Branch-and-Bound Pruning,
- Monte-Carlo Δειγματοληψία,
- Ευριστική Αναζήτηση

Στην πράξη, δεν είναι λίγες οι φορές όπου ορισμένοι *online* αλγόριθμοι ανήκουν σε παραπάνω από μια κατηγορία. Χαρακτηριστικό αποτελεί το παράδειγμα του *POMCP* αλγορίθμου, που θα συζητηθεί αναλυτικά στην ακόλουθη υποενότητα, ο οποίος βασίζεται σε Monte-Carlo δειγματοληψία, ενώ ταυτόχρονα χρησιμοποιεί ευριστική αναζήτηση προκειμένου να κατευθύνεται άμεσα σε πολλά υποσχόμενες περιοχές, εντός του δέντρου $\mathcal{T}$. Για περισσότερη εμβάθυνση περί *online* αλγορίθμων, ένα καλό σημείο εκκίνησης αποτελεί το [21].

2.3 Αλγόριθμοι Επίλυσης POMDPs με καλή κλιμάκωση

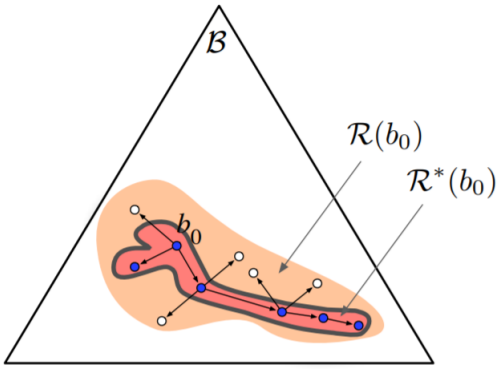
Προβλήματα του πραγματικού κόσμου, όταν μοντελοποιούνται ως POMDPs, διαθέτουν συνήθως πολύ μεγάλα domains. Υπό το καθεστώς αυτό, προβάλλει επιτακτική η ανάγκη συζήτησης αλγορίθμων, οι οποίοι υπερνικούν το υφιστάμενο υψηλό υπολογιστικό κόστος και μπορούν να παράγουν μια πολύ καλή πολιτική, έστω και προσεγγιστικά. Ειδικότερα, στην ενότητα αυτή θα ασχοληθούμε συγκεκριμένα με τρείς προσεγγιστικούς αλγορίθμους, όπου έπειτα από μελέτη και εκτενή πειραματισμό, προέκυψαν ότι είναι οι πιο ανταγωνιστικοί. Ειδικότερα πρόκειται για τον *SARSOP*, τον *POMCP* και τον *DESPOT* αλγόριθμο, με τον πρώτο να ανήκει στην οικογένεια των *point-based*, ενώ οι υπόλοιποι δύο αλγόριθμοι ανήκουν στην οικογένεια των *online* αλγορίθμων.

2.3.1 Αλγόριθμος SARSOP

Όπως έχει προαναφερθεί, η βασική ιδέα των *point-based* αλγορίθμων είναι ότι δειγματοληπτούν τον $\mathcal{B}$, ώστε να προκύψει ένα σέτ από beliefs, το οποίο θα τον αντιπροσωπεύει, κατά την προσπάθεια προσέγγισης της value function. Οι κλασικοί *point-based* αλγόριθμοι, δειγματοληπτούν μόνο το $\mathcal{R}(b_0)$, που αποτελέι υποσύνολο των προσβάσιμων beliefs, έχοντας ως σημείο αφετηρίας κάποιο αρχικό belief $b_0 \in \mathcal{B}$. Αντιθέτως ο αλγόριθμος *SARSOP* (Successive Approximations of the Reachable Space under Optimal Policies) επιχειρεί να δειγματοληπτεί κοντά στο $\mathcal{R}^*(b_0)$, το οποίο αποτελεί ένα υποσύνολο από προσβάσιμα beliefs κάτω από την υιοθέτηση βέλτιστων ενεργειών, ξεκινώντας από κάποιο b_0. Συνήθως το $\mathcal{R}^*(b_0)$ είναι αρκετά μικρότερο του $\mathcal{R}(b_0)$, ενώ ισχύει ότι $\mathcal{R}^*(b_0) \subseteq \mathcal{R}(b_0)$ (βλ. *Σχήμα 2.3*).

Θεωρητικά αποτελέσματα [11] δείχνουν, ότι προσεγγιστικές λύσεις των POMDPs μπορούν να υπολογιστούν αποδοτικά, όταν το $\mathcal{R}(b_0)$ έχει μικρό covering αριθμό[7]. Δυστυχώς όμως, αυτή η υπόθεση δεν ισχύει τις περισσότερες φορές στην πράξη. Δεδομένου ότι το $\mathcal{R}^*(b_0)$ μπορεί να είναι αρκετά μικρότερο από το $\mathcal{R}(b_0)$, προβάλλει πιθανότερο να ισχύει η υπόθεση για τό $\mathcal{R}^*(b_0)$, παρά για το $\mathcal{R}(b_0)$. Και πάλι όμως, το πρόβλημα παραμένει δύσκολο παρά αυτήν την χαλάρωση.

[7] Ο $\delta-$covering αριθμός ενός συνόλου $\mathcal{S}$, $\mathcal{C}(\delta)$, είναι ο ελάχιστος αριθμός από κύκλους ακτίνας δ, οι οποίοι χρειάζονται, ώστε να καλυφθεί το $\mathcal{S}$.



Σχήμα 2.3: Χώρος καταστάσεων belief $\mathcal{B}$, χώρος προσβάσιμων beliefs $\mathcal{R}(b_0)$ και χώρος βέλτιστων προσβάσιμων beliefs $\mathcal{R}^*(b_0)$

Με βάση τις ιδέες που παρουσιάζονται στο [11] και υποθέτοντας την ύπαρξη κάποιας βέπτιστης πολιτικής π^*, προκύπτει ότι ο υπολογισμός προσεγγιστικών λύσεων σε POMDPs είναι αρκετά δύσκολος. Όμως το πρόβλημα εύρεσης κάποιας προσεγγιστικής λύσης, γίνεται ευκολότερο όταν δίνεται ένας κατάλληλος $\delta-$covering αριθμός του $\mathcal{R}_{\pi^*}(b_0)$. Ως αποτέλεσμα, ο αλγόριθμος *SARSOP* επικεντρώνεται στην εύρεση ενός cover του $\mathcal{R}_{\pi^*}(b_0)$, προσεγγιστικά μέσω της δειγματοληψίας που πραγματοποιεί.

Ειδικότερα ο αλγόριθμος *SARSOP* βασίζεται σε τρείς επιμέρους μεθόδους:

- την *Sample*, η οποία είναι υπεύθυνη για την ευφυή δειγματοληψία του χώρου καταστάσεων belief $\mathcal{B}$,
- την *Backup*, η οποία ενσαρκώνει τον τελεστή *backup* της (2.39),
- και την *Prune*, η οποία απομακρύνει διανύσματα από το σύνολο Γ, τα οποία κυριαρχούνται από άλλα διανύσματα του συνόλου.

Είναι φανερό ότι οι μέθοδοι *Sample* και *Backup* αποτελούν τα δύο χαρακτηριστικά βήματα, με βάση τα οποία φανερώνεται η *point-based* φύση του αλγορίθμου (βλ. *Αλγόριθμο 6*).

Ταυτόχρονα, κατά την διάρκεια εκτέλεσης της μεθόδου, διατηρείται και ένα δέντρο καταστάσεων belief , αντίστοιχο με εκείνο που χρησιμοποιείται στους *online* αλγορίθμους, με μόνη διαφορά ότι δεν επιλέγεται η εύρεση όλων των προσβάσιμων, από την ρίζα, belief. Προκειμένου ο αλγόριθμος *SARSOP* να δειγματοληπτεί κοντά στο $\mathcal{R}^*(b_0)$, διατηρείται ένα άνω $\bar{V}$ και κάτω φράγμα $\underline{V}$ της βέλτιστης value function V^*, εισάγοντας μια μορφή μεροληψίας, η οποία επιρεάζει άμεσα τις μελλοντικές περιηγήσεις στο δέντρο $\mathcal{T}$, και κατ' επέκταση κατεύθυνση κίνησης στον χώρο $\mathcal{B}$. Επιπρόσθετα, το σύνολο Γ εμπεριέχει τα $\gamma-$διανύσματα που αντιπροσωπεύουν προσεγγιστικά την value function, ενώ ταυτόχρονα, έαν αρχικοποιηθεί καταλλήλως, αποτελεί και ένα καλό πάνω φράγμα της. Όσον αφορά το κάτω φράγμα της V, ο αλγόριθμος *SARSOP* χρησιμοποιεί την Sawtooth προσέγγιση [10].

Ο αλγόριθμος *SARSOP* επιστρέφει την καλύτερη πολιτική που βρέθηκε, εντός ενός προκαθορισμένου χρονικού διαστήματος. Καθώς εκτελείται επαναληπτικά, επιτυγχάνει την ελλάτωση του αρχικού κενού ϵ μεταξύ του άνω και κάτω φράγματος της value function στο b_0, έως ότου είτε το ϵ να βρεθεί κάτω από ένα προκαθορισμένο κατώφλι, είτε να έχει εξαντληθεί το προκαθορισμένο χρονικό διάστημα εκτέλεσης της μεθόδου.

Ακολούθως παρουσιάζεται συνοπτικά σε ψευδογλώσσα ο αλγόριθμος *SARSOP* (**Αλγόριθμος 8**).

Algorithm 8 SARSOP

1: Αρχικοποιήστε το σύνολο Γ με κάποια αρχικά $\gamma-$διανύσματα, ώστε να αποτελεί άνω φράγμα της V^*
2: Αρχικοποιήστε το κάτω φράγμα της V^*
3: Προσθέστε το αρχικό belief b_0 ώς μοναδικό κόμβο-ρίζα στο δέντρο καταστάσεων belief $\mathcal{T}$
4: **do**
5: $Sample(\mathcal{T}, \Gamma)$
6: Επιλέξτε ένα υποσύνολο κόμβων S από το δέντρο και για τον κάθε κόμβο $b \in S$, εφαρμόστε τον τελεστή $backup(\Gamma, b)$
7: $Prune(\mathcal{T}, \Gamma)$
8: **while** *Κριτήριο Τερματισμού Ψευδές*
9: **return** Γ

Ο αλγόριθμος *SARSOP* επιστρέφει το τελικό σύνολο $\gamma-$διανυσμάτων, Γ, με βάση το οποίο προσεγγίζεται η value function. Τα διανύσματα του συνόλου Γ διαμοιράζουν τον χώρο σε περιοχές απόφασης, όπου κάθε μία περιοχή αντιστοιχεί σε κάποια ενέργεια $u \in \mathcal{U}$. Δεδομένου ότι πρόκειται για *offline* αλγόριθμο, η σχηματιζόμενη πολιτική έχει ήδη δημιουργεί και αρκεί κάθε μελλοντικό belief να αναγνωρίζεται αρχικά σε ποία περιοχή ανήκει καί έπειτα να εκτελείται η αντίστοιχη ενέργεια. Για πιο διεξοδική επισκόπηση της μεθόδου, ο αναγνώστης παραπέμπεται σε [15, 11].

2.3.2 Αλγόριθμος POMCP

Η μέθοδος *POMCP* (Partially Observable Monte-Carlo Planning) ανήκει στην κατηγορία των *online* αλγορίθμων και συνδυάζει την Monte-Carlo ανανέωση της κατάστασης belief του *decision-maker*, με την Monte-Carlo αναζήτηση του δέντρου $\mathcal{T}$, σε κάθε προσομοίωση, έχοντας ως αφετηρία την τρέχουσα κατάσταση belief. Εξαιτίας της φιλοσοφίας στην οποία στηρίζεται ο αλγόριθμος, το δέντρο $\mathcal{T}$ που χρησιμοποιείται, διαφέρει ελαφρώς από εκείνο που χρησιμοποιούν οι κλασικές *online* μέθοδοι (βλ. 2.2.3). Ειδικότερα πρόκειται για ένα δέντρο αναζήτησης ιστορικών, το οποίο απαρτίζεται από δύο είδη κόμβων. Θεωρώντας τα επίπεδα του δέντρου $\mathcal{T}$, από το ανώτερο[8] προς το κατώτερο επίπεδο και εναλλάξ, πρόκειται για κόμβους-ιστορικά, έστω το σύνολο των οποίων απαρτίζει το $\mathcal{T}(H)$, ενώ οι κόμβοι των ενδιάμεσων επιπέδων αποτελούν κόμβους-ενέργειες[9], έστω το σύνολο των οποίων $\mathcal{T}(\mathcal{U})$ (βλ. *Σχήμα 3.2*).

Το βασικό χαρακτηριστικό της μεθόδου *POMCP* είναι ότι δεν απαιτεί την εκ των προτέρων γνώση των υφιστάμενων κατανομών πιθανότητας, οι οποίες χαρακτηρίζουν το POMDP, καθώς χρησιμοποιείται ένας POMDP προσομοιωτής. Πρόκειται ουσιαστικά για ένα black box, το οποίο επιτρέπει στην μέθοδο να αποκτά ορισμένη εμπειρία, μέσω προσομοίωσης. Δοθέντος του γεγονότος ότι ο POMDP κόσμος είναι ιδιαίτερα περίπλοκος, θεωρούμε κατ' ελάχιστον ότι μπορούμε να προσομοιώσουμε την συμπεριφορά του, με την βοήθεια του POMDP προσομοιωτή. Κατά συνέπεια, επιδιώκεται η εκμετάλλευση αυτής της γνώσης-εμπειρίας, έτσι ώστε ο *decision-maker* να δρά όσο το δυνατό καλύτερα.

[8] εκεί όπου ανήκει το αρχικό history h_0.

[9] Θεωρώντας έναν κόμβο-ιστορικό $h \in \mathcal{T}(\mathcal{H})$ και επιλέγοντας κάποια $\tilde{u} \in \mathcal{U}$, ο αντίστοιχος κόμβος-ενέργεια που αντιστοιχεί στο ιστορικό h, έπειτα από την εκτέλεση της ενέργειας $\tilde{u}$, θα συμβολίζεται ως $h\tilde{u} \in \mathcal{T}(\mathcal{U})$.

Στόχος του αλγορίθμου *POMCP* είναι να βοηθήσει τον *decision-maker*, ώστε να επιλέξει εκείνη την ενέργεια u, η οποία ελαχιστοποιεί το $V(hu)$[10], δεδομένου ότι την τρέχουσα χρονική στιγμή έχει σχηματιστεί το ιστορικό h. Δοθέντος του γεγονότος ότι ο ακρίβης υπολογισμός του $V(h)$ είναι υπολογιστικά επίπονος, προβάλλει επιτακτική η ανάγκη εύρεσης κάποιας καλής προσέγγισης αυτού. Ο αλγόριθμος *POMCP* διαχειρίζεται το συγκεκριμένο λεπτό ζήτημα, επιλέγοντας να προσεγγίσει το $V(hu)$, μέσω του προσδιορισμού των $V(huh')$ για πολλά και διαφορετικά επακόλουθα ιστορικά $h', \forall u \in \mathcal{U}$.

Προκειμένου να αποφευχθεί η περιήγηση σε περιοχές του $\mathcal{T}$, οι οποίες δεν είναι πολλά υποσχόμενες, ο αλγορίθμος *POMCP* "οχυρώνεται" με δύο τρόπους. Αρχικά όταν ένας κόμβος $h'u \in \mathcal{T}(\mathcal{U})$ επισκέπτεται για πρώτη φορά, εκτελουνταί αρκετά ***Rollouts***[11], με βάση τα οποία παρέχεται άμεσα μία πρώτη εικόνα, για το υποδέντρο με κορυφή το hu. Επιπλέον χρησιμοποιείται ο αλγόριθμος *UCB1* ([13]), προκειμένου να γίνεται αποδοτικά η περιήγηση στο δέντρο ιστορικών $\mathcal{T}$. Συνεπώς, πέρα από την Monte-Carlo προσομοίωση που πραγματοποιείται στο πλαίσιο των *Rollouts*, ο αλγόριθμος *POMCP* χρησιμοποιεί και έναν αλγόριθμο *ευρυστικής αναζήτησης*, ενισχύοντας επιπλέον την αποτελεσματικότητα του.

Λόγω του γεγονότος ότι η απόδοση του αλγορίθμου *POMCP* εξαρτάται άμεσα από το πλήθος των simulations, έχει προταθεί ([16]) μια βελτιωμένη εκδοχή του, η οποία χρησιμοποιεί την ευρυστική *RAVE* (Rapid Action Value Estimate) μέθοδο, επιτυγχάνοντας καλύτερη απόδοση από την κλασική *POMCP* μέθοδο.

Ακολούθως παρουσιάζεται, σε μορφή ψευδογλώσσας, ο κλασικός αλγόριθμος *POMCP* (**Αλγόριθμος 9**).

Algorithm 9 POMCP

function *Search(h,* b_0*)*
do
  if $hIsEmpty() == True$ **then**
    $x \sim b_0$
  else
    $x \sim B(h)$
  end if
  $Simulate(x, h, 0)$
while $TimeIsOut() == True$
return $\arg\min_{u \in \mathcal{U}} V(hu)$
end function

function *Rollout*$(x, h, depth)$
if $depth < \epsilon$ **then**
  return 0
end if
$u \sim \pi_{Rollout}(h)$
$(x', y, c) \sim \mathcal{G}(x, u)$
return $c + \rho\ Rollout(x', huy, depth + 1)$
end function

function *Simulate(x, h, depth)*
if $depth < \epsilon$ **then**
  return 0
end if
if $h \notin \mathcal{T}$ **then**
  for all $u \in \mathcal{U}$ **do**
    $\mathcal{T}(hu) \leftarrow (N_{initial}(hu), V_{initial}(hu), \emptyset)$
  end for
  return $Rollout(x, h, depth)$
end if
$u \leftarrow \arg\min_{u \in \mathcal{U}} V(hu) + c\sqrt{\frac{logN(h)}{N(hb)}}$
$(x', y, c) \sim \mathcal{G}(x, u)$
$reward \leftarrow c + \rho\ Simulate(x', huy, depth + 1)$
$B(h) \leftarrow B(h) \cup \{x\}$
$N(h) \leftarrow N(h) + 1$
$N(hu) \leftarrow N(hu) + 1$
$V(hu) \leftarrow V(hu) + \frac{reward - V(hu)}{N(hu)}$
return reward
end function

[10] Σε αυτό το σημείο το $V(hu)$ υποδηλώνει το αναμενόμενο συσσωρευτικό κόστος με το οποίο ζημειώνεται ο *decision-maker*, για όλα τα ιστορικά που ξεκινούν από το $hu \in \mathcal{T}(\mathcal{U})$.

[11] Πρόκειται για μια σειρά προσομοιώσεων, με την βοήθεια του generator $\mathcal{G}$, έχοντας ως σημείο αφετηρίας το $h'u$, μέχρι κάποιο επιθυμητό βάθος, προκειμένου το $V(h'u)$ να αρχικοποιηθεί με μια σχετικά καλή προσέγγιση.

Στον *Αλγόριθμο 9* το $B(h)$ αντιπροσωπεύει ένα δείγμα καταστάσεων, έπειτα από την παρατήρηση του ιστορικού h. Επιπλέον το $V(hu)$ αντιστοιχεί στο μέσο συσσωρευτικό κόστος, με βάση όλες τις μέχρι πρότινος προσομοιώσεις, έχοντας ως σημείο έναρξης το hu, ενώ το $N(hu)$ αντιπροσωπεύει τον αριθμό που έχει επισκεφθεί ο συγκεκριμένος κόμβος hu[12]. Ακόμη, τα $N_{initial}, V_{initial}$ ισούται με μηδέν, εκτός εάν υπάρχει κάποιου είδους προγενέστερη γνώση, η οποία θα επιβάλλει μια διαφορετική τιμή.

Τέλος η μέθοδος που παρουσιάστηκε στον *Αλγόριθμο 9*, πραγματοποείται σε κάθε βήμα απόφασης, δηλαδή όταν ο *decision-maker* καλείται να λάβει μία απόφαση, με στόχο να επιλεγεί η καλύτερη δυνατή. Μεταξύ διαδοχικών εκτελέσεων της μεθόδου, το προυπάρχουν δέντρο ιστορικών $\mathcal{T}$ διατηρείται, προκειμένου να εκμεταλλεύεται το υπολογιστικό κόστος που δαπανήθηκε, στο πλαίσιο προηγούμενων αναζητήσεων. Για λεπτομερέστερη επισκόπηση του αλγόριθμου, θεωρητικές εγγυήσεις σχετικά με την ασυμπτωτική σύγκλισή του στην βέλτιστη πολιτική αλλά και ενδεικτικά αποτελέσματα από την χρήση του σε διάφορα POMDPs με πολύ μεγάλο domain, ο αναγνώστης παραπέμπεται σε [22].

2.3.3 Αλγόριθμος DESPOT

Ο αλγόριθμος *DESPOT* (Determinized Sparse Partially Observable Tree) αποτελεί έναν *online* αλγόριθμο και βασίζεται στο δέντρο $\mathcal{T}$ που χρησιμοποιούν οι μέθοδοι της συγκεκριμένης οικόγενειας (βλ. 2.2.3). Η ειδοποιός διαφορά του αλγορίθμου *DESPOT*, έναντι άλλων *online* αλγορίθμων, είναι ότι δεν κατασκευάζει το δέντρο $\mathcal{T}$ επακριβώς, προσδιορίζοντας όλα τα προσβάσιμα δυνατά beliefs από κάποιο b_0. Αντίθετα, χρησιμοποιεί μία sparse προσέγγιση του $\mathcal{T}$, το DESPOT $\tilde{\mathcal{T}}$.

Ειδικότερα, ο αλγόριθμος *DESPOT* είναι συνυφασμένος με την έννοια του *scenario*. Ένα *scenario* για μια κατάσταση belief b, είναι μια σειρά από $\phi = (x_0, \phi_1, \phi_2, ...)$, με την κατάσταση x_0 να δειγματοληπτείται τυχαία με βάση το b και τα $\phi_i \sim U(0, 1)$. Τα *scenarios* χρησιμοποιούνται αφενός για να αναπαραστήσουν την κατάσταση belief ως ένα σύνολο από particles, αλλά και αφετέρου για να καθορίζουν την επιλογή της ακόλουθης κατάστασης και παρατήρησης για το x_0.

Όπως προαναφέρθηκε, το DESPOT $\tilde{\mathcal{T}}$ αποτελεί μια sparse προσέγγιση του $\mathcal{T}$, όπου κάθε κόμβος του περιέχει ένα σέτ από *scenarios*. Ως κόμβος-ρίζα του $\tilde{\mathcal{T}}$ είναι η αρχική κατάσταση belief b_0, ενώ από ένα δεδομένο belief b επισκέπτονται όλες οι δυνατές ενέργειες. Αντίθετα, οι παρατηρήσεις προς επίσκεψη, από έναν κόμβο-ενέργεια του $\tilde{\mathcal{T}}$, καθορίζονται από *scenarios* που χρησιμοποιούνται.

Στο [28] προτείνονται διάφορες παραλλαγές της μεθόδου *DESPOT*, οι οποίες διαφοροποιούνται ως προς τον τρόπο επιλογής της επόμενης ενέργειας από μία δεδομένη κατάσταση b. Η επικρατέστερη εκδοχή της μεθόδου *DESPOT*[13] είναι εκείνη που πραγματοποιεί *anytime ευρυστική αναζήτηση*. Κατ' αυτόν τον τρόπο, το $\tilde{\mathcal{T}}$ κατασκευάζεται σταδιακά, ακολουθώντας τα τρία βασικά βήματα:

- *Εμπρόσθια Αναζήτηση*: Η μέθοδος έχει ως σημείο έναρξης τον κόμβο-ρίζα του $\tilde{\mathcal{T}}$, b_0, και αναζητά ένα μονοπάτι, ώστε να παριηγηθεί εντός του δέντρου. Σε κάθε επισκεπτόμενη κατάσταση belief, επιλέγεται βέλτιστα κάποια ενέργεια και παρατήρηση, με βάση κάποια *heuristics*, τα οποία βασίζονται σε ένα άνω και κάτω φράγμα.
- *Αρχικοποίηση των φύλλων του* $\tilde{\mathcal{T}}$: Κατά την επίσκεψη σε ένα φύλλο του δέντρου,

[12] Παρομοίως και για το $N(h)$.

[13] ως προς θέμα απόδοσης.

b_f η μέθοδος επεκτείνεται κάποιο επιπλέον βάθος χρησιμοποιώντας όλες τις δυνατές ενέργειες, καθώς και τις παρατηρήσεις που προκύπτουν με βάση τα *scenarios* που έχουν επισκεφθεί το b_f. Έπειτα αρχικοποιεί το άνω και κάτω φράγμα των νέων κόμβων που δημιουργήθηκαν, με την εκτέλεση ενός μεγάλου αριθμού Monte-Carlo προσομοιώσεων.

- *Backup*: Έπειτα από την δημιουργία των νέων κόμβων, η μέθοδος διασχίζει το μονοπάτι αντίθετα, ανανεώνοντας ταυτόχρονα την τιμή των άνω και κάτω φραγμάτων, κάθε κόμβου του μονοπατιού.

Ο αλγόριθμος *DESPOT* επαναλαμβάνει αυτά τα τρία βήματα, έως ότου είτε το αρχικό κενό μεταξύ του άνω και κάτω φράγματος στο b_0 να είναι μικρότερο από ένα προκαθορισμένο κατώφλι, είτε εξαντληθεί κάποιο μέγιστο χρονικό διάστημα.

Δεδομένου ότι η μέθοδος είναι ιδιαίτερα περίπλοκη, είναι επιθυμητό να μην συνεχιστεί η συζήτηση περαιτέρω λεπτομερειών, καθώς κάτι τέτοιο υπερβαίνει το σκοπό της συγκεκριμένης εργασίας. Για εκτενέστερη επισκόπηση του αλγορίθμου ο αναγνώστης παραπέμπεται στο [28].

Τέλος ιδιαίτερο ενδιαφέρον παρουσιάζει η παράλληλη υβριδική[14] μέθοδος *DESPOT*, *HyP-DESPOT* ([5]). Προκύπτει ότι η μέθοδος *HyP-DESPOT* παρουσιάζει αρκετά καλύτερη απόδοση έναντι της απλής μεθόδου. Τέλος, στην περίπτωση όπου το POMDP που μελετάται έχει αρκετά μεγάλο χώρο παρατηρήσεων, αλλά ταυτόχρονα και χώρο καταστάσεων, τότε είναι προτιμότερος ο αλγόριθμος *DESPOT-*α, οποίος στηρίζεται στην δομή *DESPOT-*α, μια ελαφρώς τροποποιημένη εκδοχή της δομής *DESPOT*. Την καλύτερη επίδοση παρουσιάζει η μέθοδος *HyP-DESPOT-*α, η οποία αποτελεί την παραλληλοποιημένη μορφή της μεθόδου *DESPOT-*α. Για αναλυτικότερη επισκόπηση επί των συγκεκριμένων θεμάτων, ο αναγνώστης παραπέμπεται στο [9].

[14] Αναφέρεται ώς παράλληλη υβριδική μέθοδος, καθώς συνδυάζει την CPU και GPU παραλληλοποίηση, προκειμένου να επιτευχθεί σχεδόν real-time online planning, σε POMDPs με πολύ μεγάλο αριθμό καταστάσεων, ενεργειών ή/και παρατηρήσεων.

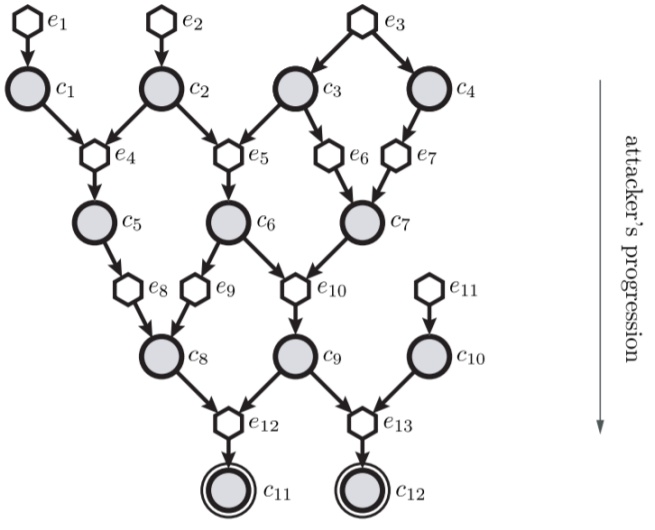
3. ΓΡΑΦΗΜΑΤΑ ΕΞΑΡΤΗΣΗΣ & ΑΝΤΙΜΕΤΩΠΗΣΗ ΑΠΕΙΛΩΝ ΕΝΤΟΣ ΔΙΚΤΥΩΝ

3.1 Εισαγωγή

Στο προηγούμενο κεφάλαιο παρουσιάστηκε λεπτομερώς η ιδέα του POMDP, οι βασικές αρχές που το διέπουν καθώς και ορισμένοι δημοφιλείς (προσεγγιστικοί ή μη) αλγόριθμοι επίλυσής του. Σε αυτό το κεφάλαιο, θα εστιάσουμε την προσοχή μας σε μια αρκετά ενδιαφέρουσα εφαρμογή, η οποία στηρίζεται στην ιδέα του POMDP και αποτελεί το επίκεντρο της συγκεκριμένης εργασίας.

Πρόκειται για το βέλτιστο δυνατό περιορισμό ενός κακόβουλου χρήστη, ο οποίος κινείται εντός κάποιου δικτύου μεγάλου μεγέθους, σε πραγματικό χρόνο. Προφανώς, ο αμυνόμενος επιθυμεί να λαμβάνει αποφάσεις με βάση κάποια πολιτική π, όσο πιο κοντά στην βέλτιστη πολιτική π^* γίνεται. Αυτό το πρόβλημα υπεράσπισης του δικτύου μοντελοποιείται ως ένα POMDP, όπου ο *decision-maker* είναι ο αμυνόμενος, ο οποίος βάσει ορισμένων παρατηρήσεων που διαθέτει, από το περιβάλλον του, καλείται να επιλέξει την τοπικά βέλτιστη ενέργεια u_l^*, ελπίζοντας ότι με την πάροδο του χρόνου θα καταφέρει να ακολουθήσει κάποια πολιτική κοντά στην βέλτιστη.

Με βάση το [17], η αναπαράσταση του εκάστοτε δικτύου στηρίζεται σε ένα είδος γραφήματος επίθεσης, ελαφρώς διαφοροποιημένου από το κλασικό, το λεγόμενο *γράφημα εξάρτησης*, προκειμένου να επιτυγχάνεται η αποτελεσματική αναπαράσταση αρκετά μεγάλων δικτύων. Στο πλαίσιο της συγκεκριμένης εργασίας, για κάθε μελετούμενο δίκτυο, θεωρείται δεδομένο το αντίστοιχο *γράφημα εξάρτησης*, χρησιμοποιώντας τα κατάλληλα εργαλεία, όπως για παράδειγμα το TVA ([12]). Ένα σχετικά απλό *γράφημα εξάρτησης* απεικονίζεται ακολούθως, όπου τα *SCs* αποτελούν συσκευές ζωτικής σημασίας του δι-



Σχήμα 3.1: Απλό *γράφημα εξάρτησης*, το οποίο αποτελείται από 12 *security conditions (SCs)* και 13 *exploits*

κτύου, ενώ *exploits* εντοπίζονται κατά μήκος κάθε δυνατής διαδρομής που μπορεί να διασχίσει ο επιτιθέμενος, μεταξύ διαδοχικών *SCs*. Κατ' αυτόν τον τρόπο, δίνεται η δυνατότητα στον *decision-maker* (αμυνόμενος) να λαμβάνει αποφάσεις κάθε χρονική στιγμή, οι οποίες αντιστοιχούν σε "μπλοκάρισμα" συγκεκριμένων *exploits* του δικτύου, με στόχο τον περιορισμό της εξάπλωσης του επιτιθέμενου.

Όπως είναι εμφανές και από το *Σχήμα 3.1*, τα *γραφήματα εξάρτησης* αναπαρίστανται ως άκυκλοι υπεργράφοι $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, με $\mathcal{N} = \{c_1, ..., c_{|\mathcal{N}|}\}$ [1] και $\mathcal{E} = \{e_1, ..., e_{|\mathcal{E}|}\}$ [2]. Επιπλέον, υπάρχει ένα σύνολο *goal SCs*, έστω $\mathcal{N}_{goal} \subseteq \mathcal{N}$[3], όπου εάν ο επιτιθέμενος καταφέρει να τα "καταλάβει", τότε θεωρείται ότι έχει επιτύχει τον στόχο του. Σκοπός λοιπόν είναι, κατ' ελάχιστον να προφυλαχθούν τα $c_i \in \mathcal{N}_{goal}$ του εκάστοτε *γραφήματος εξάρτησης* και έαν είναι δυνατό επιδιώκεται να προστατευθεί ολόκληρο το δίκτυο με το λιγότερο δυνατό κόστος, πράγμα που σηματοδοτεί ότι είναι επιθυμητή η όσο το δυνατό συντομότερη ανακοπή της πορείας του επιτιθέμενου (σε υψηλότερο επίπεδο του γραφήματος), εντός του δικτύου.

Ως φυσικό επακόλουθο, λοιπόν, πρωταρχική ανάγκη αποτελεί ο προσδιορισμός του επιπέδου ασφάλειας του δικτύου, πράγμα που είναι ισοδύναμο με τον βαθμό εξάπλωσης του επιτιθέμενου, μέσα σε αυτό. Για τον λόγο αυτόν, ορίζεται ως κατάσταση ασφαλείας ένα σύνολο από *SCs*, τα οποία έχει "καταλάβει" ο επιτιθέμενος (ενεργό *SCs*). Προφανώς ο *decision-maker*, δεν δύναται να γνωρίζει επάκριβως ποια *SCs* έχουν καταληφθεί από τον επιτιθέμενο. Παραμόνο, λαμβάνοντας ορισμένες θορυβώδεις παρατηρήσεις από τις κινήσεις του επιτιθέμενου, κάθε χρονική στιγμή, μπορεί να συντηρεί μια "πίστη", σχετικά με το ποιοί κόμβοι μάλλον έχουν καταληφθεί.

Μία κατάσταση ασφάλειας, θεωρείται ότι είναι εφικτή όταν κάθε ένα ενεργό *SCs* της, έχει ενεργοποιηθεί από κάποιο, μη μπλοκαρισμένο προφανώς, *exploit*, του οποίου όλα τα *pre-SCs* και *post-SCs*, είναι ενεργά. Η κατάσταση ασφάλειας εξελίσσεται, προφανώς, στοχαστικά με την πάροδο του χρόνου, συναρτήσει των ενεργειών που υιοθετούν αμυνόμενος και επιτιθέμενος. Στο πλαίσιο του προβλήματος *αμυνόμενου - επιτιθέμενου* θεωρείται ότι κάθε χρονική στιγμή, δρα πρώτα ο αμυνόμενος, αφού έχει λάβει κάποια σχετική πληροφορία σχετικά με τις τελευταίες κινήσεις του επιτιθέμενου και έπειτα ο τελευταίος χρησιμοποιεί τις τρέχουσες δυνατότητες τους, έχοντας ως σκοπό την βαθύτερη εξάπλωσή του εντός του δικτύου. Εάν καταφέρει να καταλάβει τουλάχιστον ένα νέο *SC*, τότε τροποποιείται η τρέχουσα κατάσταση ασφάλειας. Προκειμένου να μειωθεί σημαντικά ο αριθμός των καταστάσεων ασφαλείας, γίνεται υπόθεση ύπαρξης *μονοτονικότητας* (*monotonicity assumption*). Κατα συνέπεια, γίνεται η παραδοχή πως τα *SCs* που έχει καταλάβει ο επιτιθέμενος, σε κάποια χρονική στιγμή, παραμένουν από κει και πέρα αποκλειστικά υπό τον έλεγχο του. Η αλήθεια είναι ότι πρόκειται για μία ιδιαίτερα περιοριστική υπόθεση, όμως με την βοήθεια της, γίνεται εφικτή η μοντελοποίηση ενός τόσο σύνθετου προβλήματος.

Από την λεπτομερή συζήτηση που γίνεται στο [18], στο οποίο στηρίζεται το συγκεκριμένο κεφάλαιο, προκύπτει τελικά ότι τα dynamics του προβλήματος *αμυνόμενου - επιτιθέμενου* μπορούν να περιγραφούν ως POMDP, όπου στον ρόλο του *decision-maker* είναι ο αμυνόμενος, ενώ ο επιτιθέμενος απαρτίζει το περιβάλλον του, πράγμα που σηματοδοτεί ότι είναι υπεύθυνος για την εκπομπή των παρατηρήσεων που λαμβάνει ο αμυνόμενος, σχετικά με τις τελευταίες επιδιώξεις του. Λαμβάνοντας υπόψη, ότι σε κάθε χρονική στιγμή ενδέχεται να τροποποιούνται η "πίστη" του *decision-maker* είτε για την τρέχουσα κατάσταση ασφάλειας είτε τον τύπο του επιτιθέμενου[4], είτε και για τα δύο, καθίσταται σαφές ότι η κατάσταση belief θα προσπαθεί να προσδιορίσει, με την πάροδο του χρόνου, αμφότερα τις καταστάσεις ασφάλειας καθώς και τον τύπο του επιτιθέμενου, με βάση την ληγθείσα πληροφορία. Προφανώς, λοιπόν, η κατάσταση belief για το συγκεκριμένο POMDP μο-

[1] το σύνολο των *SCs.*

[2] το σύνολο των *exploits*.

[3] Για παράδειγμα, στο *Σχήμα 3.1* είναι φανερό ότι $\mathcal{N}_{goal} = \{c_{11}, c_{12}\}$.

[4] Στο [18] γίνεται η υπόθεση ότι ο αμυνόμενος γνωρίζει εκ των προτέρων μια γκάμα πιθανών διαφορετικών τύπων επιτιθέμενων και ανάλογα με τις πρόσφατες ληφθείσες παρατηρήσεις (*security alerts*) από το περιβάλλον του, έχει την δυνατότητα να μεταβάλει την τρέχουσα ”πίστη” του, σχετικά με τον τύπο του επιτιθέμενου που έχει να αντιμετωπίσει.

ντέλο, βασίζεται στο ιστορικό πληροφοριών που έχει στην διάθεση του ο *decision-maker*, $H_k = \{b_0, u_0, y_1, ..., u_{k-1}, y_k\}$ και θα πέρνει την μορφή

$$b_k = \begin{bmatrix} b_k^{1,1} & b_k^{1,2} & ... & b_k^{1,|\Phi|} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ b_k^{|\mathcal{X}|,1} & b_k^{|\mathcal{X}|,2} & ... & b_k^{|\mathcal{X}|,|\Phi|} \end{bmatrix} \in \mathcal{B}_{\mathcal{X}\times\Phi} \tag{3.1}$$

με $b_k^{i,l} = \mathbb{P}(x_k = i, \phi_k = l | H_k = h)$ και το Φ υποδηλώνει το πεπερασμένο σύνολο πιθανών τύπων επιτιθέμενων, ενώ το $\mathcal{B}_{\mathcal{X}\times\Phi}$ υποδηλώνει τον χώρο Simplex του πεπερασμένου χώρου $\mathcal{X}\times\Phi$, όπου $\mathcal{B}_{\mathcal{X}\times\Phi} \subset \mathcal{X}\times\Phi$.

Αντίστοιχα με την (2.5), προκύπτει

$$b_k(x_k, \phi_k) = T(b_{k-1}, u_{k-1}, y_k) = \mathbb{P}(x_k, \phi_k | b_{k-1}, u_{k-1}, y_k), \text{ όπου } T(b_{k-1}, u_{k-1}, y_k) = \tag{3.2}$$

$$= \frac{\left\{\sum_{x_{k-1}\in\mathcal{X}, \phi_{k-1}\in\Phi} b_{k-1}(x_{k-1}, \phi_{k-1}) P^{\phi_{k-1}}_{x_{k-1},x_k}(u_{k-1}) q_{\phi_{k-1},\phi_k}\right\} \cdot}{\sum_{\tilde{x}_k\in\mathcal{X}, \tilde{\phi}_k\in\Phi}[\left\{\sum_{x_{k-1}\in\mathcal{X}, \phi_{k-1}\in\Phi} b_{k-1}(x_{k-1}, \phi_{k-1}) P^{\phi_{k-1}}_{x_{k-1},\tilde{x}_k}(u_{k-1}) q_{\phi_{k-1},\tilde{\phi}_k}\right\} \cdot}$$

$$\frac{\cdot \left\{\sum_{\tilde{x}_{k-1}\in\mathcal{X}, \tilde{\phi}_{k-1}\in\Phi} b_{k-1}(\tilde{x}_{k-1}, \tilde{\phi}_{k-1})\ O^{\tilde{\phi}_{k-1}}_{\tilde{x}_{k-1},x_k,y_k}(u_{k-1})\right\}}{\cdot \left\{\sum_{\tilde{x}_{k-1}\in\mathcal{X}, \tilde{\phi}_{k-1}\in\Phi} b_{k-1}(\tilde{x}_{k-1}, \tilde{\phi}_{k-1})\ O^{\tilde{\phi}_{k-1}}_{\tilde{x}_{k-1},\tilde{x}_k,y_k}(u_{k-1})\right\}]} \tag{3.3}$$

Τα συστατικά στοιχεία του POMDP το οποίο χρησιμοποιείται στο πλαίσιο του προβλήματος *αμυνόμενου - επιτιθέμενου* είναι

$$P^{l,m}_{i,j}(u) = \mathbb{P}(x_k = j, \phi_k = m | x_{k-1} = i, \phi_{k-1} = l, u_{k-1} = u) \tag{3.4}$$

$$= \mathbb{P}(\phi_k = m | \phi_{k-1} = l)\ \mathbb{P}(x_k = j | x_{k-1} = i, \phi_{k-1} = l, u_{k-1} = u) = q_{l,m}\ P^l_{i,j}(u),$$

$$O^l_{i,j,y}(u) = \mathbb{P}(y_k = y | x_k = j, x_{k-1} = i, u_{k-1} = u, \phi_{k-1} = l), i, j \in \mathcal{X},\ l \in \Phi,\ y \in \mathcal{Y} \tag{3.5}$$

$$c(x_k, \phi_k, u_k) = \omega c_x(x_k, \phi_k) + (1-\omega) c_u(u_t),\ \text{με } 0 \le c_u(\tilde{u}) < \infty, \forall \tilde{u} \in \mathcal{U} \tag{3.6}$$

Όσον αφορά την (3.6), επιλέγεται μία τέτοια μορφή κόστους, προκειμένου να καθίσταται δυνατός ο προσδιορισμός του *trade-off* μεταξύ της ασφάλειας και της διαθεσιμότητας του δικτύου, ανάλογα με την περίσταση. Οι δύο αυτές έννοιες είναι προφανώς ανταγωνιστικές, ιδιαίτερα όταν αναφερόμαστε σε δίκτυα μεγάλου μεγέθους, τα οποία εξυπηρετούν εκατομμύρια χρήστες και τυχαίνει κάποια στιγμή να δέχονται μία κακόβουλη επίθεση, σε κάποιο τμήμα τους. Για αναλυτικότερη επισκόπηση σχετικά με την επιλογή της συγκεκριμένης μορφής κόστους, ο αναγνώστης παραπέμπεται στην ενότητα E του [18].

3.2 Πρόβλημα Αμυνόμενου - Επιτιθέμενου

Δοθέντος του POMDP που παρουσιάστηκε στην προηγούμενη ενότητα, το οποίο χαρακτηρίζεται από τις (3.4) - (3.6) και είναι ειδικά προσαρμοσμένο στις ανάγκες του προβλήματος *αμυνόμενου - επιτιθέμενου*, βασικότερος στόχος είναι ο *decision-maker* να λαμβάνει

κάθε χρονική στιγμή την βέλτιστη αμυντική ενέργεια, λαμβάνοντας υπόψην του το *trade-off* μεταξύ ασφάλειας και διαθεσιμότητας του δικτύου, καθώς και τις κινήσεις του επιτιθέμενου. Η αμυντική ενέργεια που υιοθετείται κάθε χρονική στιγμή, προκύπτει από το mapping $\pi : \mathcal{B}_{\mathcal{X}\times\Phi} \rightarrow \mathcal{U}$, ενώ το κριτήριο βάσει του οποίου λαμβάνονται οι αποφάσεις, επιλέγεται να είναι η ελαχιστοποίηση του discounted αναμενόμενου κόστους, με το οποίο θα επιβαρυμθεί ο *decision-maker* σε άπειρο χρονικό ορίζοντα, ξεκινώντας από κάποιο αρχικό belief b_0,

$$\pi^* = \arg\min_{\pi} \mathbb{E}_{\pi} \left\{ \sum_{k=0}^{\infty} \rho^k r(x_k, \phi_k, \pi(x_k)) | b_0 \right\}, b_0 \in \mathcal{B}_{\mathcal{X}\times\Phi} \text{ και} \tag{3.7}$$

$$r(x_k, \phi_k, \pi(x_k)) = \sum_{x_k \in \mathcal{X}, \phi \in \Phi} b_k^{x_k,\phi_k} c(x_k, \phi_k, u_k) \tag{3.8}$$

Ως γνωστών, εύρεση της βέλτιστης λύσης αυτού του προβλήματος, αποτελέι την βέλτιστη αμυντική πολιτική $\forall b \in \mathcal{B}_{\mathcal{X}\times\Phi}$ και με αυτόν τον τρόπο συνεπάγεται η υιοθέτηση ενεργειών, προκειμένου να επιτευχθεί το επιθυμητό *trade-off* που έχει καθοριστεί με την ανάλογη επιλογή του ω στην (3.6).

3.3 Προσαρμοσμένος αλγόριθμος POMCP για το πρόβλημα Αμυνόμενου - Επιτιθέμενου

Έχοντας ως προσανατολισμό προβλήματα του πραγματικού κόσμου, γίνεται εύκολα αντιληπτό ότι το συσχετιζόμενο POMDP που θα χρησιμοποιηθεί, προκειμένου να μοντελοποιηθεί το πρόβλημα, αναμένεται να είναι αρκετά μεγάλου μεγέθους. Ως συνέπεια, επιδυώκεται η επιλογή κάποιου προσεγγιστικού αλγορίθμου, ο οποίος θα είναι ικανός να επιλυεί σε πραγματικό χρόνο το πρόβλημα *αμυνόμενου - επιτιθέμενου*, παράγωντας μια αρκετά καλή τοπική πολιτική κάθε χρονική στιγμή, έτσι ώστε ο *decision-maker* να δρά με σχεδόν βέλτιστον τρόπο, σε βάθος χρόνου.

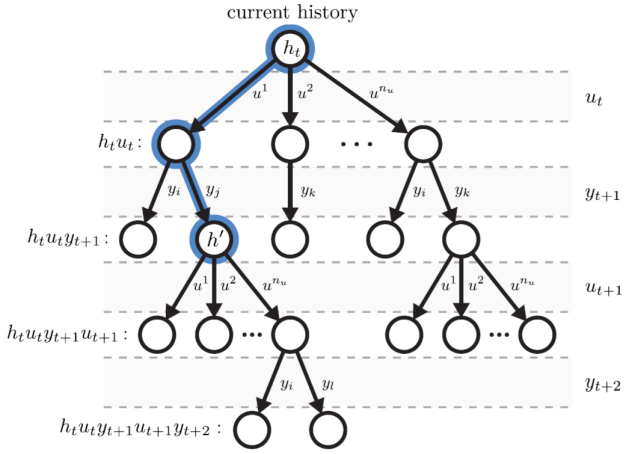
Στο [18] παρουσιάζεται ο αλγόριθμος *POMCP*, ελαφρώς τροποποιημένος, ως η καλύτερη δυνατή επιλογή (βλ. 2.3.2), προκειμένου να αντιμετωπίζεται αυτοματοποιημένα, σε πραγματικό χρόνο, οποιαδήποτε απειλή εντοπίζεται στο δίκτυο. Ως γνωστόν, ο αλγόριθμος *POMCP* αποτελείται από δύο βασικά βήματα: *i)* την επιλογή της καλύτερης δυνατής ενέργειας που μπορεί να υιοθετηθεί και *ii)* το belief update[5].

Ως προς την επιλογή της u_l^*, ο προσαρμοσμένος αλγόριθμος *POMCP* για το πρόβλημα *αμυνόμενου - επιτιθέμενου* ταυτίζεται με τον κλασικό *POMCP* αλγόριθμο. Κατ' αυτό τον τρόπο, ξεκινάν να δειγματοληπτούνται ζευγάρια (x, ϕ), με βάση τον πίνακα από beliefs, b_k, κάποια χρονική στιγμή k. Έπειτα εκτελείται η μέθοδος *Simulate* του *POMCP* αλγορίθμου, η οποία βασίζεται σε Monte-Carlo προσομοιώσεις (*Rollouts*), καθώς και στο *heuristic UCB1*, προκειμένου να πραγματοποιείται πιο συχνά περιήγηση σε περιοχές του δέντρου ιστοριών, οι οποίες είναι αρκετά υποσχόμενες , όπως συζητήθηκε αναλυτικά στην αντίστοιχη υποενότητα.

Έπειτα από την υιοθέτηση της καλύτερης δυνατής ενέργειας u_l^*, λαμβάνονται ορισμένα *security alerts*, σχετικά με τον τρόπο με τον οποίο αντέδρασε ο επιτιθέμενος. Στην συνέχεια, πραγματοποιείται *update* του ιστορικού, όπως παρουσιάζεται στο ακόλουθο σχήμα, ενώ ταυτόχρονα είναι αναγκαίο το *update* της κατάστασης belief. Στο σημείο αυτό, ο τροποποιημένος POMCP αλγόριθμος διαφοροποιείται από τον κλασικό *POMCP*, εκμεταλ-

[5]Ο *exact* υπολογισμός του *belief update* της (3.3) (ή αντίστοιχα της (2.5)) είναι ιδιαίτερα δύσκολος υπολογιστικά και ως συνέπεια επιλέγεται αντ' αυτού, ένα ευφυές προσεγγιστικό σχήμα.

λευόμενος τον τρόπο με τον οποίο δημιουργούνται τα *security alerts*, με βάση την πίστη του *decision-maker* σχετικά με την πρόοδο του attacker εντός του δικτύου. Ειδικότερα, ο



Σχήμα 3.2: Ένα στιγμιότυπο ενός δέντρου ιστορικών

αμυνόμενος διατηρεί κάποια προσεγγιστική κατάσταση belief κάθε χρονική στιγμή, έστω B_k, στην οποία εμπεριέχονται ένας προκαθορισμένος αριθμός ζευγαριών (x, ϕ), που είναι γνωστά ως *particles*. Σε αντίθεση με τον κλασικό *POMCP* αλγόριθμο[6], η προσαρμοσμένη εκδοχή του εκμεταλεύεται τον τρόπο με τον οποίο παράγονται τα *security alerts*, καθώς είναι αρκετά πιθανό το δειγματολειπτημένο alert, στην έξοδο του $\mathcal{G}$, να μην συμπίπτει συχνά με το πραγματικό y[7], πράγμα που θα σηματοδοτούσε σημαντικές επιπρόσθετες σημαντικές καθιστερίσεις.

Ειδικότερα, ας θεωρήσουμε ότι την στιγμή k μια δειγματοληπτημένη κατάσταση ασφαλείας και ο αντίστοιχος τύπος του attacker, στην έξοδο του $\mathcal{G}$, είναι κάποια συγκεκριμένα $x_k \in \mathcal{X}$, $\phi_k \in \Phi$, αντίστοιχα. Ανάλογα με τα διαθέσιμα *exploits*, έστω το σύνολο των οποίων είναι $\mathcal{E}(x_k)$, δομείται μια γκάμα από αναμενόμενα alerts προς εμφάνιση στην επόμενη προσομοίωση, έστω $\mathcal{Y}_{x_k} = \cup_{e \in \mathcal{E}(x_k)} \mathcal{O}(e)$. Παρόλα αυτά, με θετική πιθανότητα μπορούν να δημιουργηθούν *false alarms*, για αρκετούς λόγους[8], πράγμα που ισοδυναμεί με την παραγωγή alerts, τα οποία δεν παρέχουν στον αμυνόμενο πληροφορία, σχετική με τις πραγματικές κινήσεις του επιτιθέμενου. Όσον αφορά το *belief update*, στον προσαρμοσμένο *POMCP* κάθε προκύπτον alert, ελέγχεται εάν ανήκει στο $\mathcal{Y}_{x_k}$ και όχι με όλα τα δυνατά alerts, $\mathcal{P}(\mathcal{Y})$. Έστω το binary διάνυσμα $o \in \mathcal{Y} = \{0,1\}^{|\mathcal{Y}|}$, όπου έχει 1 στην i-οστή θέση, σε περίπτωση όπου το i-οστό alert έχει εντοπιστεί. Εαν το $\mathcal{I}(o(\mathcal{Y}_{x_k})) \in \mathcal{Y}_{x_k}$, τότε γίνεται αποδεκτό με πιθανότητα

$$p_{yes}(x_k, \phi_k) = \left\{ \prod_{i \in \mathcal{I}(o(\mathcal{Y}^c_{x_k})=1)} \zeta_i(\phi_k) \right\} \cdot \left\{ \prod_{i \in \mathcal{I}(o(\mathcal{Y}^c_{x_k})=0)} (1 - \zeta_i(\phi_k)) \right\} \tag{3.9}$$

όπου το $\mathcal{I}(o(\mathcal{Y}^c_{x_k}))$ σηματοδοτεί τους δείκτες των θέσεων του διανίσματος o που δεν ανήκουν στο $o(\mathcal{Y}_{x_k})$, ενώ $\zeta_i(\phi_k)$ την πιθανότητα ψευδούς εκπομπής του i-οστού alert, συναρ-

[6]Χρησιμοποιεί τον generator $\mathcal{G}$, προκειμένου να συγκεντρώσει ένα επαρκές πλήθος ζευγαριών ($\tilde{x}$, y), όπου το y είναι η παρατήρηση του περιβάλλοντος, την δεδομένη χρονική στιγμή. Σε περίπτωση όπου το δειγματοληπτημένο alert δεν ταυτίζεται με το y, τότε τα αντίστοιχα $\tilde{x}$ απλώς αγνοούνται.

[7]ιδιαίτερα σε μοντέλα ασφάλειας με αρκετά μεγάλο χώρο παρατηρήσεων.

[8]Εάν κάτι τέτοιο γίνεται σκοπίμως, τότε στόχος είναι η παραπλάνηση του *decision-maker*. Διαφορετικά, *false alarms* μπορούν να προκύψουν από σφάλματα στο σύστημα αισθητήρων του δικτύου, το οποίο είναι υπεύθυνο για την συλλογή alerts που σχετίζονται με τις τελευταίες κινήσεις του επιτιθέμενου.

τήσει το τύπου του attacker ϕ_k. Η *normalized* μορφή της (3.9) παίρνει την μορφή

$$\tilde{p}_{yes}(x_k, \phi_k) = \frac{p_{yes}(x_k, \phi_k)}{\eta}, \text{ με } \eta = \max_{(x,\phi) \in B_{k-1}} p_{yes}(x, \phi) \tag{3.10}$$

πράγμα που εξασφαλίζει ότι, σε αυτή την περίπτωση, τα *particles* γίνονται αποδεκτά πιο συχνά, κατά την φάση του *belief update*, έναντι του κλασικού αλγορίθμου *POMCP*.

Σε μορφή ψευδογλώσσας, παρουσιάζεται η μέθοδος του *belief update* του τροποποιημένου *POMCP* αλγορίθμου (**Αλγόριθμος 10**).

Algorithm 10 Belief Update για Προσαρμοσμένο POMCP Αλγόριθμο

Αρχικοποιήστε $numberOfParticles$,
$B_k \leftarrow \{\emptyset\}, succAdds \leftarrow 0$

function *NewBeliefUpdate*(B_{k-1}, u_{k-1}, y_k):
for $i = 1$ to $numberOfParticles$ **do**
  $(x, \phi) \sim B_{k-1}$
  $(x', \phi', y', -) \sim \mathcal{G}(x, \phi, u_{k-1})$
  if $\mathcal{I}(y'(\mathcal{Y}_x)) = \mathcal{I}(y_k(\mathcal{Y}_x))$ **then**
    $B_k \leftarrow B_k \cup \{x', \phi'\}$ // με πιθανότητα $\tilde{p}_{yes}(x, \phi)$
    $succAdds \leftarrow sucAdds + 1$
  end if
end for
end function

4. ΣΤΟΧΑΣΤΙΚΗ ΠΡΟΣΕΓΓΙΣΗ ΚΑΙ ΕΝΙΣΧΥΤΙΚΗ ΜΑΘΗΣΗ

4.1 Κίνητρο

Στο πλαίσιο του συγκεκριμένου κεφαλαίου θα συζητηθεί η βασική ιδέα που διέπει τους *stochastic gradient* αλγορίθμους, οι οποίοι συγκλίνουν σε κάποιο τοπικό βέλτιστο, με στόχο τον καθορισμό μιας καλής *randomized* πολιτικής. Δεδομένου ορισμένων δυσκολιών που θα παρουσιαστούν αναλυτικά στην συνέχεια, καθίσταται αδύνατη η εφαρμογή ενός exact *stochastic gradient* σχήματος. Ως συνέπεια, καταφεύγουμε στην χρήση *gradient* εκτιμητών, ευελπιστώντας να δώσουν μια καλή προσέγγιση.

Βασικός στόχος της συγκεκριμένης εργασίας είναι η παρουσίαση της μεθόδου *score function gradient* εκτίμησης (*SFGE*) (βλ. 4.3.4, 4.6), στο πλαίσιο προσέγγισης μια καλής *off-line global* πολιτικής για το πρόβλημα του *αμυνόμενου - επιτιθέμενου*, το οποίο παρουσιάστικε αναλυτικά στο τρίτο κεφάλαιο.

Από την πλευρά μας, δόθηκε ιδιαίτερη προσοχή στην οικογένεια των *stochastic gradient* αλγορίθμων, δεδομένου ότι είναι υπολογιστικά εφικτοί, σε αντίθεση με τον *exact* δυναμικό προγραμματισμό. Ταυτόχρονα, αποτελούν την βάση της ενισχυτικής μάθησης, καθώς δεν απαιτείται προγενέστερη γνώση παραμέτρων που χαρακτηρίζουν τα MDPs ή POMDPs, αφού υπάρχει δυνατότητα παρατήρησης και εν τέλει εκμάθησης του υφιστάμενου στοχαστικού περιβάλλοντος. Ουσιαστικά πρόκειται για *simulation-based* προσεγγιστικούς αλγορίθμους, όπου με την πάροδο του χρόνου αποκτούν γνώση, πράγμα που συνεπάγεται την καλύτερη προσαρμογή τους στις εκάστοτε συνθήκες.

4.2 Εισαγωγικές Έννοιες

Ας θεωρήσουμε, προς στιγμήν, το πρόβλημα ενός MDP άπειρου ορίζοντα με discounted κόστος

$$\pi^* = \arg\min_{\pi} \mathbb{E}_{\pi} \left\{ \sum_{k=0}^{\infty} \rho^k c(x_k, \pi(x_k)) \right\} \tag{4.1}$$

στοχεύοντας στην εύρεση μιας αρκετά καλής πολιτικής, χωρίς να ακολουθηθεί το μονοπάτι της ακριβούς επίλυσης του προβλήματος, μέσω του δυναμικού προγραμματισμού. Έστω ότι κάθε χρονική στιγμή k επιλέγεται μια ενέργεια σύμφωνα με την παραμετροποιημένη πολιτική $\pi_\theta(x)$, όπου π_θ κάποια προκαθορισμένη συνάρτηση που παραμετροποιείται από ένα διάνυσμα $\theta \in \mathbb{R}^p$. Στόχος είναι ο υπολογισμός του

$$\theta^* = \arg\min_{\theta} C(\theta), \text{ με } C(\theta) = \mathbb{E}_{\pi_\theta} \left\{ \sum_{k=0}^{\infty} \rho^k c(x_k, \pi_\theta(x_k)) \right\} \tag{4.2}$$

ο οποίος μπορεί να επιτευχθεί, χρησιμοποιώντας τον *stochastic gradient* αλγόριθμο της μορφής

$$\theta_{n+1} = \theta_n - \epsilon_n \widehat{\nabla_\theta}\, C_n(\theta_n),\ n = 0, 1, 2, ... \tag{4.3}$$

όπου με $\widehat{\nabla_\theta}\, C_n(\theta_n)$ υποδηλώνεται μία προσέγγιση του $\nabla_\theta C(\theta)$ στο θ_n[1]. Ταυτόχρονα, όπως και με όλες τις μεθόδους που εμπίπτουν στην οικογένεια των *gradient* αλγορίθμων, αναμένεται η σύγκλιση σε κάποιο τοπικό στάσιμο σημείο της (4.1).

[1] Το $\widehat{\nabla_\theta}\, C_n(\theta_n)$ θα υπολογιστεί προφανώς με βάση τα παρατηρήσιμα $\{c(x_k, \theta), k = 1, 2, ...\}$, καθώς είναι η μοναδική διαθέσιμη πληροφορία.

Γενικά μιλώντας, έστω μία Μαρκοβιανή διαδικασία $\{x_k\}_{k=0}^{N}$, με πίνακα μετάβασης P_θ, όπου είναι *εργοδική*. Ακολούθως παρουσιάζοντα δύο βασικοί τύποι simulated-based *gradient* εκτιμητών για τέτοιου είδους MP

- *Finite Difference Gradient Εκτιμητές*: Δεν απαιτούν κάποια γνώση του πίνακα μετάβασης της MP, αλλά δυστυχώς υποφέρουν από το $bias - variancetrade - off$. Οι δύο κλασικότεροι αλγόριθμοι της συγκεκριμένης οικογένειας είναι ο FDSA και SPSA.
- *Gradient Εκτιμητές που εκμεταλλεύονται τον* P_θ: Ένας από τους αλγορίθμους που ανήκουν στην συγκεκριμένη ομάδα είναι και ο *score function gradient* εκτιμητής (*SFGE*), στον οποίο και θα επικεντρωθούμε για το υπόλοιπο της παρούσας εργασίας.

Ειδικότερα, στις ακόλουθες υποενότητες θα συζητηθεί αναλυτικά η μορφή του *SFGE* στο πλαίσιο των Μαρκοβιανών διαδικασιών (MP), των Μαρκοβιανών διαδικασιών αποφάσεων (MDP), των μερικώς παρατηρούμενων Μαρκοβιανών διαδικασιών αποφάσεων (POMDP) καθώς και εν τέλει του προβλήματος *αμυνόμενου - επιτιθέμενου*.

4.3 Score Function Gradient Εκτίμηση (SFGE)

4.3.1 SFGE για MP

Σε αυτήν την υποενότητα θα συζητηθεί αναλυτικά ο *score function stochastic gradient* αλγόριθμος, για το πλαίσιο των Μαρκοβιανών διαδικασιών. Η βασική ιδέα που θα παρουσιαστεί, αποτελεί τροχοπέδη για την κατάλληλη προσαρμογή του συγκεκριμένου αλγορίθμου, στο πλαίσιο των POMDPs και του προβλήματος *αμυνόμυνου-επιτιθέμενου* (βλ. Κεφάλαιο 3), όπως θα συζητηθεί στις υποενότητες που ακολουθούν.

Στόχος είναι ο υπολογισμός του $\theta^* \in \Theta \subset \mathbb{R}^p$ το οποίο ελαχιστοποιεί το αναμενόμενο κόστος

$$C(\theta) = \mathbb{E}_{d_\theta}\{c(x)\} = \sum_{x \in \mathcal{X}} c(x)\, d_\theta(x) \tag{4.4}$$

Δυστυχώς, τις περισσότερες φορές είτε η στάσιμη κατανομή της Μαρκοβιανής διαδικασίας d_θ είτε το κόστος $c(x)$ δέν είναι εκ των προτέρων γνωστά[2]. Συνεπώς, είναι αδύνατο να υπολογιστεί το $C(\theta)$ στην (4.4), επακριβώς. Παρόλο αυτά, μπορούν να αποκτηθούν θορυβώδεις παρατηρήσεις $\{c(x_k, \theta), k = 1, 2, ...M\}, \forall \theta \in \Theta$, προκειμένου να υπολογιστεί το $\widehat{\nabla_\theta}\, C_n(\theta_n)$.

Έπειτα, χρησιμοποιώντας τον ευρέως διαδεδομένο και δημοφηλή αλγόριθμο στοχαστικής βελτιστοποίησης *stochastic gradient* της (4.3), υπολογίζεται το θ^* της (4.4), επαναληπτικά για κάθε χρονική στιγμή $k = 1, 2, ...$.

Υποθέτοντας ότι $\forall \theta \in \Theta$, μπορούμε να παρατηρήσουμε θορυβώδεις trajectories από κόστη $c(x_k), k = 1, 2, .., M$, μέσω προσομοιώσεων με μηδενική μέση τιμή θορύβου, άμεσος στόχος, πλέον, είναι η εκτίμηση του $\nabla_\theta \mathbb{E}_{d_\theta}\{c(x)\} = \nabla_\theta (c^T d_\theta)$.

Προφανώς καθώς $M \to \infty$, ανεξάρτητα από το αρχικό d_0, ισχύει

$$\lim_{M \to \infty} c^T (P_\theta^M)^T d_0 = c^T d_\theta \tag{4.5}$$

[2]Στην περίπτωση μάλιστα της στάσιμης κατανομής d_θ, ακόμα και ο P_θ να είναι γνωστός, ο υπολογισμός της είναι ιδιαίτερα δύσκολος και υπολογιστικά επώδυνος.

και επομένως

$$\nabla_\theta(c^T d_\theta) \approx \nabla_\theta(c^T (P_\theta^M)^T d_0) = d_0^T \ (\nabla_\theta P_\theta^M) \ c \tag{4.6}$$

$$= d_0^T \ \left\{ \sum_{k=0}^{M-1} P_\theta^{M-k-1} \ (\nabla_\theta P_\theta) \ P_\theta^k \right\} \ c \tag{4.7}$$

Πέρνοντας την *finite sample* προσέγγιση της (4.8) προκύπτει

$$\nabla_\theta(c^T d_\theta) \approx \mathbb{E}_{x_0:x_M} \left\{ c(x_M) \sum_{k=1}^{M} \frac{\nabla_\theta P_{x_{k-1},x_k;\theta}}{P_{x_{k-1},x_k;\theta}} \right\} \tag{4.8}$$

Ορίζοντας το S_M^θ

$$S_M^\theta = \sum_{k=1}^{M} \frac{\nabla_\theta P_{x_{k-1},x_k;\theta}}{P_{x_{k-1},x_k;\theta}} \tag{4.9}$$

είναι φανερό ότι μπορεί να υπολογιστεί επαναληπτικά, ως εξής

$$S_k^\theta = \frac{\nabla_\theta P_{x_{k-1},x_k;\theta}}{P_{x_{k-1},x_k;\theta}} + S_{k-1}^\theta, \text{ με } k = 2, ..., M \text{ και}$$

$$S_1^\theta = \frac{\nabla_\theta P_{x_0,x_1;\theta}}{P_{x_0,x_1;\theta}} \tag{4.10}$$

δίνεται ακολούθως ο αλγόριθμος *SFGE* για Μαρκοβιανές διαδικασίες, σε μορφή ψευδογλώσσας (**Αλγόριθμος 11**).

Algorithm 11 Score Function Gradient Εκτίμηση για MP

1: Simulate την Μαρκοβιανή διαδικασία $\{x_0, ..., x_M\}$ με πίνακα μετάβασης P_θ
2: $S_1^\theta \leftarrow \frac{\nabla_\theta P_{x_0,x_1;\theta}}{P_{x_0,x_1;\theta}}$
3: **for** $k = 2$ to M **do**
4: $\quad S_k^\theta \leftarrow \frac{\nabla_\theta P_{x_{k-1},x_k;\theta}}{P_{x_{k-1},x_k;\theta}} + S_{k-1}^\theta$
5: **end for**
6: $\widehat{\nabla_\theta} \ C_M(\theta) \leftarrow \frac{1}{M} \sum_{k=1}^{M} c(x_k) S_k^\theta$
7: **return** $\widehat{\nabla_\theta} \ C_M(\theta)$

4.3.2 SFGE για MDP

Στην συγκεκριμένη υποενότητα θα περιγραφεί αναλυτικά ο *SFGE* αλγόριθμος, ειδικά προσαρμοσμένος για Μαρκοβιανές διαδικασίες αποφάσεων, που έχει ως στόχο την αναζήτηση στο *policy space* για την έρεση μιας καλής πολιτικής.

Ειδικότερα, προκειμένου να επεκταθεί η ιδέα που παρουσιάστηκε αναλυτικά στην προηγούμενη υποενότητα, θεωρούμε ένα *unichain MDP* μέσου κόστους (βλ. 1.2.4, 1.2.5). Επίσης, έστω μία νέα Μαρκοβιανή διαδικασία με καταστάσεις $z_k = (x_k, u_k)$, όπου $x_k \in \mathcal{X}, u_k \in \mathcal{U}$. Είναι εύκολο να δειχθεί ότι ο πίνακας μετάβασης της $\{z_k\}$ δίνεται από

$$P_{(i,u),(j,\tilde{u})} = \mathbb{P}(x_{k+1} = j, u_{k+1} = \tilde{u} | x_k = i, u_k = u) = \theta_{j,\tilde{u}} P_{i.j}(u) \tag{4.11}$$

όπου $\theta_{j,\tilde{u}} = \mathbb{P}(u_{k+1} = \tilde{u} | x_{k+1} = j)$

Ας θεωρηθεί στο σημείο αυτό η στάσιμη κατανομή της Μαρκοβιανής διαδικασίας $\{z_k\}$, $d_\theta(i,a)$ με $i \in \mathcal{X}$ και $a \in \mathcal{U}$. Επιπλέον, έστω ότι δεν είναι γνωστές οι πιθανότητες μετάβασης, $P_{i,j}(u)$, του υφιστάμενου MDP, πράγμα που σηματοδοτεί ότι καθίσταται αδύνατος ο υπολογισμός της $d^*_\theta(i,a)$, επιλύοντας το αντίστοιχο πρόβλημα γραμμικού προγραμματισμού των (1.30) - (1.32), το οποίο παρουσιάζεται στο *Θεώρημα 1.5 (βλ. 1.2.5)*.

Αντίθετα, είναι δυνατό να επιλυθεί το ακόλουθο ισοδύναμο πρόβλημα γραμμικού προγραμματισμού, ώστε να προσδιοριστεί, εν τέλει, η βέλτιστη *randomized* πολιτική

$$\theta^*_{i,a}(\psi) = \mathbb{P}(action = a | state = i\,;\psi), i \in \mathcal{X}, a \in \mathcal{U}$$

η οποία πλέον παραμετροποιείται από ένα διάνυσμα παραμέτρων ψ. Το πρόβλημα βελτιστοποίησης τροποποιείται ως εξής

$$\min_{\psi \in \Psi} C(\psi), \text{ με } C(\psi) = \mathbb{E}_{d_{\theta(\psi)}}\{c(x,u)\} \tag{4.12}$$

$$\text{s.t. } \theta \in \Theta$$

Γίνεται εμφανές ότι άμεσος στόχος είναι ο προσδιορισμός του βέλτιστου ψ^*, προκειμένου να εντοπιστεί η καλύτερη δυνατή πολιτική, η οποία αντιπροσωπεύεται από την κατανομή $\theta(\psi^*)$. Για τον λόγο αυτό, θα χρησιμοποιηθεί ο *stochastic gradient* αλγόριθμος

$$\psi_{n+1} = \psi_n - \epsilon_n \, \widehat{\nabla_\psi} C_n(\theta(\psi_n)) \tag{4.13}$$

με βάση το ψ, έναντι του θ.

Επιπλέον, λόγω του ότι η θ είναι μία κατανομή πιθανότητας, είναι αναγκαίο η παραμετροποίησή της ως προς το ψ, να γίνει με ιδιαίτερη προσοχή, ώστε να εξασφαλιστεί πως $\theta \in \Theta$[3]. Στο πλαίσιο της συγκεκριμένης εργασίας, επιλέχθηκε η πιο απλή και ευρέως διαδεδομένη, εκθετική παραμετροποίηση για το θ, δηλαδή

$$\theta_{i,a}(\psi) = \frac{e^{\psi_{i,a}}}{\sum_{u \in \mathcal{U}} e^{\psi_{i,a}}}, \text{ με } \psi_{i,a} \in \mathbb{R}, i \in \mathcal{X}, a \in \mathcal{U} \tag{4.14}$$

εξασφαλίζοντας προφανώς ότι $\theta \in \Theta$.

Με βάση την μέθοδο *SFGE* για MP (*Αλγόριθμος 11*), θα παρουσιαστεί ακολούθως η προσαρμοσμένη μορφή της, για το πλαίσιο των MDPs. Ειδικότερα, από την (4.11) προκύπτει ότι

$$\nabla_\psi \, P_{(i,u),(j,\tilde{u})}(\theta(\psi)) = P_{i,j}(u) \, (\nabla_\psi \, \theta_{j,\tilde{u}}(\psi)) \tag{4.15}$$

και άμεσος στόχος είναι η εύρεση του καλύτερου δυνατού $\psi_{i,a} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{X}|}$, το οποίο θα καθορίζει την προκύπτουσα *randomized* πολιτική.

Ισχυρό πλεονέκτημα της *SFGE* μεθόδου, αποτελεί το γεγονός ότι, τελικά, δεν απαιτεί την εκ των προτέρων γνώση του $P(u)$, όπως καθίσταται σαφές από την ακόλουθη προσαρμοσμένη εκδοχή της (**Αλγόριθμος 12**).

Γίνεται αντιληπτό ότι στην συγκεκριμένη μέθοδο, χρησιμοποιείται επιπλέον μια παράμετρος $\beta \in (0,1)$, σε σχέση με την αντίστοιχη μέθοδο για MP. Κάτι τέτοιο είναι αναγκαίο, δεδομένου ότι ο *score function gradient* εκτιμητής υποφέρει από σχετικά μεγάλη διασπορά και σκοπός είναι η μείωσή της. Κατά αυτόν τον τρόπο, εισάγεται ένα είδος μνήμης κατά τον υπολογισμό των διαδοχικών $S^\psi_k, k = 1,2,...,M$.

[3] Ανεξάρτητα από την παραμετροποίηση που θα επιλεγεί, θα πρέπει να ισχύει:
$\Theta = \{\theta_{x,u} \geq 0 \text{ και } \sum_{u \in \mathcal{U}} \theta_{x,u} = 1, \text{ με } x \in \mathcal{X}, u \in \mathcal{U}\}$.

Algorithm 12 Score Function Gradient Εκτίμηση για MDP

1: **for** $n = 1$ to N **do**
2: $S_1^{\psi_{l,a}} \leftarrow \frac{\nabla_{\psi_{l,a}} P_{z_0,z_1\,;\,\theta}}{P_{z_0,z_1\,;\,\theta}}$
3: **for** $k = 1$ to $M-1$ **do**
4: $S_{k+1}^{\psi_{l,a}} \leftarrow \frac{\nabla_{\psi_{l,a}} P_{z_k,z_{k+1}\,;\,\theta}}{P_{z_k,z_{k+1}\,;\,\theta}} + \beta\, S_k^{\psi_{l,a}}$, όπου

$$\frac{\nabla_{\psi_{l,a}} P_{z_k,z_{k+1}\,;\,\theta}}{P_{z_{k-1},z_k\,;\,\theta}} = \frac{\cancel{P_{i,j}(u)}\,\nabla_{\psi_{l,a}}\theta_{l,a}(\psi)}{\cancel{P_{i,j}(u)}\theta_{l,a}(\psi)} = \begin{cases} 1-\theta_{x_{k+1},u_{k+1}} & \textit{if } l = x_{k+1}, a = u_{k+1} \\ -\theta_{x_{k+1},a} & \textit{if } l = x_{k+1}, a \in \mathcal{U} - \{u_{k+1}\} \end{cases} \tag{4.16}$$

5: **end for**
6: $\widehat{\nabla_\theta}\, C_n(\theta) \leftarrow \frac{1}{M}\sum_{k=1}^{M} c(x_k,u_k) S_k^{\psi_{l,a}}$
7: $\psi_{n+1} \leftarrow \psi_n - \epsilon_n\, \widehat{\nabla_\psi}\, C_n(\theta_n)$
8: **end for**
9: **return** ψ_N

4.3.3 SFGE για POMDP

Ακολούθως παρουσιάζεται η *SFGE* μέθοδος για μερικώς παρατηρούμενες Μαρκοβιανές διαδικασίες αποφάσεων, η οποία βασίζεται σε μια αρκετά παρόμοια ιδέα με εκείνη που χρησιμοποιήθηκε στην προηγούμενη υποενότητα.

Ειδικότερα, έστω ένα κλασικό POMDP με discounted κόστος, το οποίο διαθέτει τις ακόλουθες παραμέτρους

$$(\mathcal{X}, \mathcal{U}, \mathcal{Y}, P(u), B(u), c(u), \rho)$$

Δεδομένου ότι η *SFGE* μέθοδος ανήκει στην οικογένεια των αλγορίθμων ενισχυτικής μάθησης, είναι αναγκαίο σε αυτό το σημείο να υποτεθεί, ότι οι παράμετροι $P(u), B(u), c(u)$ δεν είναι εκ των προτέρων γνωστές. Επιπλέον, παρομοίως με την λογική που ακολουθήθηκε στην προηγούμενη υποενότητα, υπάρχει δυνατότητα παρατήρησης δειγματοληπτημένων $\{c(x_k,u_k), k = 1,2,...\}$, χωρίς να είναι γνωστή η συνάρτηση κόστους $c(x,u)$ και προφανώς η επακριβής κατάσταση x_k.

Δυστυχώς, μη γνωρίζωντας τα $P(u), B(u)$ είναι αδύνατο να υπολογιστεί επ' ακριβώς η κατάσταση belief b_k, κάθε χρονική στιγμή k, η οποία αποτελεί και επαρκής εκτιμήτρια του ιστορικού H_k. Ως εκτούτου, το H_k προσεγγίζεται χρησιμοποιώντας ένα "παράθυρο ολίσθησης" με μνήμη L, το οποίο διατηρεί τις τελευταίες L ενέργειες και παρατηρήσεις. Επιπλέον θεωρούμε την MP με καταστάσεις $z_k = (x_k, u_k, H_k^L)$, όπου $H_k^L = \{u_{k-L}, y_{k-L+1}, ..., u_{k-1}, y_k\}$. Οι πιθανότητες μετάβασης της συγκεκριμένης MP, από κάποια κατάσταση $z_k = (x_k, u_k, H_k^L)$ σε κάποια $z_{k+1} = (x_{k+1}, u_{k+1}, H_{k+1}^L)$, είναι εύκολο να προκύψει πως δίνεται από

$$P_{(i,u,h),(j,\tilde{u},\tilde{h})} = \mathbb{P}(x_{k+1} = j, u_{k+1} = \tilde{u}, H_{k+1}^L = \tilde{h} | x_k = i, u_k = u, H_k^L = h) = \theta_{u,\tilde{h}} P_{i,j}(u) O_{j,y}(u) \tag{4.17}$$

$$\text{όπου } \theta_{u,\tilde{h}} = \mathbb{P}(u_{k+1} = \tilde{u} | H_{k+1}^L = \tilde{h}), \text{ και } y_{k+1} = y$$

Η *randomized* πολιτική $\mathbb{P}(u_{k+1}|H_k^L)$ παραμετροποιείται ως προς το ψ

$$\theta_{h,a} = \mathbb{P}(action = a | history = h\,;\psi), \forall a \in \mathcal{U}, h \in \mathcal{H} \tag{4.18}$$

χρησιμοποιώντας παρομοίως εκθετική παραμετροποίηση.

Παρομοίως με την περίπτωση των MDPs, στόχος είναι ο προσδιορισμός του

$$\psi^* = \arg\min_{\theta\in\Theta'} C(\theta(\psi)), \text{ όπου } C(\theta) = \mathbb{E}_{d_\theta}\{c(x,u)\} \tag{4.19}$$

$$\text{s.t. } \Theta' = \{\theta_{h,u} \geq 0 \text{ και } \sum_{u\in\mathcal{U}} \theta_{h,u} = 1, \text{ με } h \in \mathcal{H}, u \in \mathcal{U}\}$$

όπου το d_θ υποδηλώνει την στάσιμη κατανομή του MP, με κατάστασεις $z_k = \{x_k, u_k, H_k^L\}$. Παρομοίως, το καλύτερο δυνατό ψ^* του προβλήματος (4.19), προσδιορίζεται με την βοήθεια της *stochastic gradient* μεθόδου (4.13).

Γενικότερα, ο *SFGE* αλγόριθμος για POMDPs είναι σχεδόν ίδιος με εκείνον που παρουσιάστηκε για MDPs, εκτός από την (4.16), η οποία για τα POMDPs προσαρμόζεται ως εξής

$$\frac{\nabla_{\psi_{h,a}} P_{z_k,z_{k+1};\theta}}{P_{z_k,z_{k+1};\theta}} = \frac{\cancel{P_{i,j}(a)}\,\cancel{O_{j,y}(a)}\,\nabla_{\psi_{h,a}}\theta_{h,a}(\psi_{h,a})}{\cancel{P_{i,j}(a)}\,\cancel{O_{j,y}(a)}\,\theta_{h,a}(\psi_{h,a})} = \tag{4.20}$$

$$= \begin{cases} 1-\theta_{H_{k+1}^L,u_{k+1}} & \textit{if } h = H_{k+1}^L, a = u_{k+1} \\ -\theta_{H_{k+1}^L,a} & \textit{if } h = H_{k+1}^L, a \in \mathcal{U} - \{u_{k+1}\} \end{cases}$$

Κατά τα άλλα, η μέθοδος για τα POMDPs είναι ίδια με τον *Αλγόριθμο 12* [4] και για αυτό τον λόγο κρίνεται αναγκαίο, να μην επαναληφθεί και σε αυτό το σημείο. Ιδιαίτερο ενδιαφέρον παρουσιάζει η προσαρμοσμένη μορφή της *SFGE* μεθόδου, στο πρόβλημα του *αμυνόμενου - επιτιθέμενου*, όπως παρουσιάζεται στην ακόλουθη υποενότητα.

4.3.4 SFGE για το πρόβλημα Αμυνόμενου - Επιτιθέμενου

Ακολουθώντας παρόμοια λογική με την προηγούμενη υποενότητα, στόχος πλέον είναι η παρουσιάση της *SFGE* μεθόδου για το πρόβλημα του *αμυνόμενου - επιτιθέμενου*, το οποίο παρουσιάστηκε αναλυτικά στο τρίτο κεφάλαιο.

Έστω μια νέα MP, με $z_k = (x_k, u_k, \phi_k, H_k^L)$, όπου $H_k^L = \{u_{k-L}, \phi_{k-L}, y_{k-L+1}, ..., y_k\}$. Οι πιθανότητες μετάβασής της, από μία κατάσταση $z_k = \{x_k, u_k, \phi_k, H_k^L\}$ σε μία άλλη κατάσταση $z_{k+1} = \{x_{K+!}, u_{k+1}, \phi_{K+1}, H_{k+1}^L\}$, προκύπτει ακολούθως

$$P_{(i,u,\phi,h),(j,\tilde{u},\tilde{\phi},\tilde{h})} = \mathbb{P}(x_{k+1}=j, u_{k+1}=\tilde{u}, \phi_{k+1}=\tilde{\phi}, H_{k+1}^L = \tilde{h} | x_k = i, u_k = u, \phi_k = \phi, H_k^L = h)$$

$$\begin{aligned} &= \mathbb{P}(x_{k+1}=j | x_k = i, u_k = u, \phi_k = \phi) \cdot \\ &\quad \cdot \mathbb{P}(u_{k+1}=\tilde{u}, \phi_{k+1}=\tilde{\phi}, H_{k+1}^L = \tilde{h} | x_k = i, u_k = u, x_{k+1} = j, H_k^L = h) \\ = \quad &\mathbb{P}(x_{k+1}=j | x_k = i, u_k = u, \phi_k = \phi)\, \mathbb{P}(y_{k+1}=y | x_{k+1} = j, u_k = u, \phi_k = \phi) \cdot \\ &\cdot \mathbb{P}(\phi_{k+1}=\tilde{\phi} | \phi_k = \phi)\, \mathbb{P}(u_{k+1}=\tilde{u} | H_{k+1}^L = \tilde{h}) \end{aligned} \tag{4.21}$$

Η πολιτική $\mathbb{P}(u_{k+1}|H_k^L)$ παραμετροποιείται ως προς το ψ χρησιμοποιώντας εκθετική παραμετροποίηση, όπως ακριβώς στην (4.18). Στην συγκεκριμένη περίπτωση στόχος είναι η εύρεση του καλύτερου δυνατού ψ^*, για το οποίο ισχύει

$$\psi^* = \arg\min_{\theta\in\Theta'} C(\theta(\psi)), \text{ όπου } C(\theta) = \mathbb{E}_{d_\theta}\{c(x,u,\phi)\} = \sum_{i\in\mathcal{X}}\sum_{a\in\mathcal{U}}\sum_{\phi\in\Phi}\sum_{h\in\mathcal{H}} c(i,a,\phi)\; d_\theta(i,a,\phi,h) \tag{4.22}$$

[4] Μοναδική διαφορά αποτελεί το γεγονός ότι, ο *SFGE* αλγόριθμος για το πλαίσιο των POMDPs αποτελεί *suboptimal* μέθοδο, σε αντίθεση με την περίπτωση των MDPs.

με την d_θ να υποδηλώνει την στάσιμη κατανομή της θεωρούμενης MP, με καταστάσεις $z_k = \{x_k, u_k, \phi_k, H_k^L\}$. Ακολούθως παρουσιάζεται η προσαρμοσμένη εκδοχή της *SFGE* μεθόδου, σε μορφή ψευδογλώσσας, για $L = 0$, πράγμα που συνεπάγεται ότι $H_k = \{y_k\}$. (**Αλγόριθμος 13**). Τέλος, αρκετά εύκολα μπορεί να τροποποιηθεί ο ακόλουθος αλγόριθμος, για την περίπτωση όπου $L > 0$.

Algorithm 13 Score Function Gradient Εκτίμηση για το Πρόβλημα *Αμυνόμενου - Επιτιθέμενου*

Αρχικοποιήστε τυχαία το $\psi \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{Y}|}$
for $n = 1$ to N **do**
  $S_1^{\psi_{y,a}} \leftarrow \frac{\nabla_{\psi_{l,a}} P_{z_0,z_1\,;\,\theta}}{P_{z_0,z_1\,;\,\theta}}$
  for $k = 1$ to $M - 1$ **do**
    $S_{k+1}^{\psi_{y,a}} \leftarrow \frac{\nabla_{\psi_{l,a}} P_{z_k,z_{k+1}\,;\,\theta}}{P_{z_k,z_{k+1}\,;\,\theta}} + \beta\, S_k^{\psi_{l,a}}$, όπου

$$\frac{\nabla_{\psi_{l,a}} P_{z_k,z_{k+1}\,;\,\theta}}{P_{z_{k-1},z_k\,;\,\theta}} = \frac{\nabla_{\psi_{l,a}} \theta_{l,a}(\psi)}{\theta_{l,a}(\psi)} = \begin{cases} 1 - \theta_{y_{k+1},u_{k+1}} & \textit{if } l = y_{k+1}, a = u_{k+1} \\ -\theta_{y_{k+1},a} & \textit{if } l = y_{k+1}, a \in \mathcal{U} - \{u_{k+1}\} \end{cases} \tag{4.23}$$

  end for
  $\widehat{\nabla_\theta}\, C_n(\theta) \leftarrow \frac{1}{M} \sum_{k=1}^{M} c(x_k, u_k, \phi_k) S_k^{\psi_{l,a}}$
  $\psi_{n+1} \leftarrow \psi_n - \epsilon_n\, \widehat{\nabla_\psi}\, C_n(\theta_n)$
end for
return ψ_N

ΠΑΡΑΡΤΗΜΑ Ι: ΒΕΛΤΙΣΤΟ FILTERING - HMM FILTER

Θεωρώντας ένα απλό Μαρκοβιανό μοντέλο, το οποίο χαρακτηρίζεται από την κατανομή αρχικής κατάστασης,

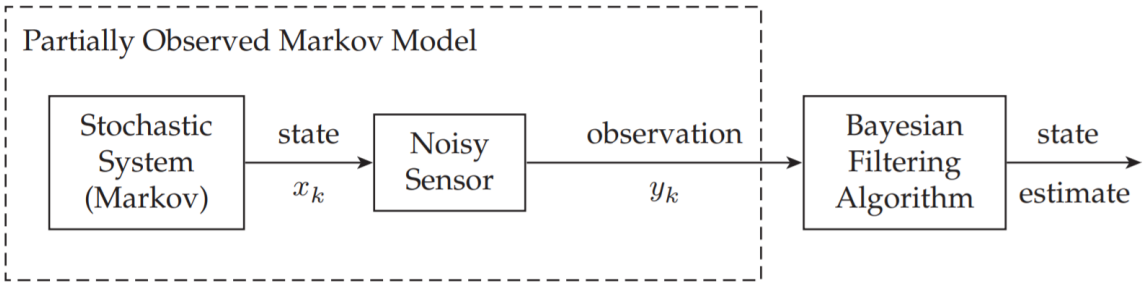
$$b_0 = [\mathbb{P}(x_0 = 1), ..., \mathbb{P}(x_0 = |\mathcal{X}|)]^T \tag{I.1}$$

καθώς επίσης και από τις κατανομές

$$P_{i,j} = \mathbb{P}(x_{k+1} = j | x_k = i), \tag{I.2}$$

$$O_{i,y} = \mathbb{P}(y_k = y | x_k = i) \tag{I.3}$$

στόχος είναι η εκτίμηση της τρέχουσας κατάστασης x_k, δεδομένου ότι N παρατηρήσεις, έστω $y_{1:N} = (y_1, ..., y_N)$, είναι διαθέσιμες. Το πρόβλημα αυτό είναι γνωστό ως *βέλτιστος προσδιορισμός κατάστασης*.



Σχήμα 1: Bayesian filtering για εκτίμηση κατάστασης

Ανάλογα με την επιλογή των k, N, διακρίνονται οι ακόλουθες τρείς περιπτώσεις:

- *Optimal Prediction:* Επιλέγοντας $k > N$, το πρόβλημα *prediction* στοχεύει στον υπολογισμό του $\hat{x}_{k|N}$, διαθέτοντας παρατηρήσεις έως την χρονική στιγμή N,
- *Optimal Filtering:* Επιλέγοντας $k = N$, το πρόβλημα ανάγεται στον προσδιορισμό του $\hat{x}_{k|k}$, δοθέντων των παρατηρήσεων του παρελθόντος και του παρόντος,
- *Optimal Smoothing:* Επιλέγοντας $k < N$, το πρόβλημα στοχεύει στον προσδιορισμό του $\hat{x}_{k|N}$, δοθέντων των παρατηρήσεων του παρελθόντος, του παρόντος και του μέλοντος εως κάποια χρονική στιγμή N.

Στο πλαίσιο αυτού του παραρτήματος, θα συζητηθεί αποκλειστικά το πρόβλημα του *βέλτιστου filtering*, αρχικά ως προς ένα απλό Μαρκοβιανό μοντέλο (I.1) - (I.3) και έπειτα στο πλαίσιο των POMDPs, με στόχο η κατάληξη στις σχέσεις (2.4) - (2.5).

Στόχος, λοιπόν, είναι ο προσδιορισμός του

$$\hat{x}_{k|k} = \mathbb{E}\{x_k | y_{1:k}\} = \sum_{x \in \mathcal{X}} x_k \, p(x_k | y_{1:k}), \text{ για } k = 1, 2, 3, ... \tag{I.4}$$

όπου το $p(x_k|y_{1:k})$ αντιπροσωπεύει την *posterior* κατανομή της κατάστασης του Μαρκοβιανού μοντέλου την χρονική στιγμή k, δοθέντων των παρατηρήσεων έως την χρονική στιγμή k. Έστω

$$b_k(x) = \mathbb{P}(x_k = x | y_{1:k}), x \in \mathcal{X} \tag{I.5}$$

Συνεπώς

$$b_k = \frac{p(x_k, y_1, ..., y_{k-1}, y_k)}{p(y_1, ..., y_{k-1}, y_k)} = \frac{p(y_k|x_k, y_{1:k-1}) \sum_{x_{k-1}\in\mathcal{X}} p(x_k|x_{k-1}) p(x_{k-1}, y_{1:k})}{\sum_{\tilde{x_k}\in\mathcal{X}} p(y_k|\tilde{x_k}, y_{1:k-1}) p(\tilde{x_k}, y_{1:k-1})}$$

$$= \frac{\cancel{p(y_{1:k-1})} p(y_k|x_k, y_{1:k-1}) \sum_{x_{k-1}\in\mathcal{X}} p(x_k|x_{k-1}) p(x_{k-1}|y_{1:k-1})}{\cancel{p(y_{1:k-1})} \sum_{\tilde{x_k}\in\mathcal{X}} p(y_k|\tilde{x_k}, y_{1:k-1}) \sum_{x_{k-1}\in\mathcal{X}} p(\tilde{x_k}|x_{k-1}) p(x_{k-1}|y_{1:k-1})}$$

$$= \frac{p(y_k|x_k, y_{1:k-1}) \sum_{x_{k-1}\in\mathcal{X}} p(x_k|x_{k-1}) b_{k-1}(x_{k-1})}{\sum_{\tilde{x_k}\in\mathcal{X}} p(y_k|\tilde{x_k}, y_{1:k-1}) \sum_{x_{k-1}\in\mathcal{X}} p(\tilde{x_k}|x_{k-1}) b_{k-1}(x_{k-1})} \tag{I.6}$$

και τελικά προκύπτει ότι $\hat{x}_{k|k} = \sum_{x\in\mathcal{X}} x_k \; b_k(x)$, όπου το b_k δίνεται από την (I.6).

Δυστυχώς, ο υπολογισμός στην (I.6) καθίσταται αδύνατος καθώς είτε το $k \to \infty$, είτε εάν ο χώρος καταστάσεων $\mathcal{X}$ δεν είναι διακριτός, όπως έχουμε υποθέσει, αλλά αντιθέτως είναι συνεχής. Απαιτείται επομένως η ύπαρξη κάποιας πεπερασμένης *sufficient statistic* της b_k. Το *Hidden Markov Model (HMM)* φίλτρο αποτελεί το ένα εκ των δύο γνωστών φίλτρων, πεπερασμένης διάστασης. Στις υπόλοιπες περιπτώσεις, η αναδρομική σχέση (I.6) υπολογίζεται μόνο προσεγγιστικά, πράγμα που σηματοδοτεί ότι τα αντίστοιχα φίλτρα είναι *sub-optimal*.

Επιλέγοντας ως Μαρκοβιανό μοντέλο το *HMM*, προκύπτει το *HMM Φίλτρο*, προκειμένου να εκτιμηθεί η κατάσταση της υφιστάμενης MP, κάθε χρονική στιγμή k. Μοναδική πληροφορία αποτελούν οι λαμβανόμενες θορυβώδεις παρατηρήσεις, μέχρι την χρονική στιγμή k. Συνεπώς οι (I.1) - (I.3) αντιπροσωπεύουν πλήρος το *HMM*. Απαιτείται επομένως η γνώση μιας αρχικής κατανομής b_0, που αποτελεί την "πίστη" μας για το ποία είναι η αρχική κατάσταση της υφιστάμενης MP, ο πίνακας μετάβασης της MP καθώς και η κατανομή που χαρακτηρίζει την εκπομπλή θορυβωδών παρατηρήσεων δεδομένου αποκλειστικά της τρέχουσας κατάστασης της MP. Ως συνέπεια, για *HMM φίλτρο*, η (I.6) παίρνει την εξής μορφή

$$b_k = T(b_{k-1}, y_k) = \frac{O_{x_k,y_k} \sum_{x_{k-1}\in\mathcal{X}} P_{x_{k-1},x_k} b_{k-1}(x_{k-1})}{\sum_{\tilde{x_k}\in\mathcal{X}} O_{\tilde{x_k},y_k} \sum_{x_{k-1}\in\mathcal{X}} P_{x_{k-1},\tilde{x_k}} b_{k-1}(x_{k-1})} \tag{I.7}$$

με $O_{i,y}, P_{i,j}$ όπως δίνονται από τις (I.2) - (I.3) και $b_k(x)$ από την (I.5).

Παρατηρώντας την (I.7) (ή αντίστοιχα την (I.6)), είναι φανερό ότι εάν μία χρονική στιγμή k είναι γνωστή η $b_{k-1}(x_{k-1}), \forall x_{k-1} \in \mathcal{X}$, καθώς και η λαμβανόμενη παρατήρηση y_k, καθίσταται εφικτή η εύρεση του b_k, εκείνη την χρονική στιγμή. Επομένως, η b_{k-1} αποτελεί επαρκή στατιστική της ιστορίας $H_{k-1} = \{y_1, ..., y_{k-1}\}$, όπου σε συνδυασμό με την y_k αρκούν, ώστε να προσδιοριστεί το νέο b_k.

Η αντίστοιχη μορφή του *HMM filtering* (I.6) για POMDPs είναι η (2.5), η οποία προκύπτει με λογική παρόμοια με εκείνη που ακολουθήθηκε στην (I.6), λαμβάνοντας υπόψη, επιπλέον, ότι το filtering πραγματοποιείται στο πλαίσιο του POMDP και συνεπώς οι (I.2) - (I.3) αντικαθίστανται από τις

$$O_{i,y}(u) = \mathbb{P}(y_k = y|x_k = i, u_{k-1} = u), \text{ και} \tag{I.8}$$

$$P_{i,j}(u) = \mathbb{P}(x_k = j|x_{k-1} = i, u_{k-1} = u) \tag{I.9}$$

ΑΝΑΦΟΡΕΣ

[1] R. Bellman. *Dynamic Programming*. Princeton University Press, 1st edition edition, 1957.

[2] D. Bertsekas. *Dynamic Programming and Optimal Control (Vol I)*. Nashua, NH: Athena Scientific, 3rd edition edition, 2007.

[3] D. Bertsekas. *Dynamic Programming and Optimal Control (Vol II)*. Nashua, NH: Athena Scientific, 3rd edition edition, 2007.

[4] D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[5] P. Cai, Y. Luo, D. Hsu, and W. Lee. HyP-Despot: A Hybrid Parallel Algorithm for Online Planning under uncertainty. *CoRR*, abs/1802.06215, 2018.

[6] A. Cassandra. *Exact and Approximate Algorithms for Partially Observed Markov Decision Process*. PhD thesis, Dept. Computer Science, Brown University, 1998.

[7] J. Pineau G. Shani and R. Kaplow. A Survey of Point-Based POMDP Solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, 2013.

[8] R. I. Brafman G. Shani and S. E. Shimony. Forward Search Value Iteration for POMDPs. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 2619–2624.

[9] N. Priyadarshini Garg, D. Hsu, and W. Lee. DESPOT-Alpha: Online POMDP Planning with Large State and Observation Spaces. In *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019.*

[10] M. Hauskrecht. Value-Function Approximations for Partially Observable Markov Decision Processes. *Journal Artificial Intelligence Research*, 13:33–94, 2000.

[11] D. Hsu, W. Lee, and N. Rong. What makes some POMDP problems easy to approximate? In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 689–696.

[12] S. Jajodia. Topological Analysis of Network Attack Vulnerability. In *Proceedings of the 2007 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2007, Singapore, March 20-22, 2007*, page 2.

[13] L. Kocsis and C. Szepesvári. Bandit Based Monte-Carlo Planning. In *Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006, Proceedings*, pages 282–293.

[14] V. Krishnamurthy. *Partially Observed Markov Decision Processes*. Cambridge University Press, 2016.

[15] H. Kurniawati, D. Hsu, and W. Lee. SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. In *Robotics: Science and Systems IV, Eidgenössische Technische Hochschule Zürich, Zurich, Switzerland, June 25-28, 2008*.

[16] P. Liu, J. Chen, and H. Liu. An improved monte carlo pomdps online planning algorithm combined with rave heuristic. In *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pages 511–515.

[17] E. Miehling, M. Rasouli, and D. Teneketzis. A POMDP Approach to the Dynamic Defense of Large-Scale Cyber Networks. *IEEE Trans. Information Forensics and Security*, 13(10):2490–2505, 2018.

[18] E. Miehling, M. Rasouli, and D. Teneketzis. A POMDP Approach to the Dynamic Defense of Large-Scale Cyber Networks. *IEEE Trans. Information Forensics and Security*, 13(10):2490–2505, 2018.

[19] C. H. Papadimitriou and J. N. Tsitsiklis. The Complexity of Markov Decision Processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[20] P. Poupart, Kee-Eung Kim, and Dongho Kim. Closing the Gap: Improved Bounds on Optimal POMDP Solutions. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling, ICAPS 2011, Freiburg, Germany June 11-16, 2011*.

[21] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online Planning Algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[22] D. Silver and J. Veness. Monte-Carlo Planning in Large POMDPs. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.*, pages 2164–2172.

[23] R. Smallwood and E. Sondik. The Optimal Control of Partially Observable Markov Processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[24] T. Smith and R. Simmons. Heuristic Serach Value Interation for POMDPs. *In Conference on uncertainty in artificial intelligence (UAI)*, 2004.

[25] T. Smith and R. Simmons. Point-Based POMDP Algorithms: Improved Analysis and Implementation. *CoRR, 2012*, 2012.

[26] E. Sondik. *The Optimal Control of Partially Observable Markov Process*. PhD thesis, Dept. Computer Science, Stanford University, 1971.

[27] R. Sutton and A. Barto. *Reinforcement Learning*. Cambridge, MA: MIT Press, 2nd edition edition, 2018.

[28] N. Ye, A. Somani, D. Hsu, and W. Lee. DESPOT: Online POMDP Planning with Regularization. *Journal of Artificial Intelligence Reasearch*, 58:231–266, 2017.