UNIVERSITAT DE LLEIDA
Escola Politècnica Superior
Grau en Enginyeria Informàtica
Models de Computació i Complexitat

Restless Bandit

Joaquim Picó Mora, Sergi Simón Balcells

PraLab2

Professorat : M.Valls

Data : Divendres 27 de Març

Contents

1 Introducció

Aquest curs, per una iniciativa de grup, hem començat a llegir diferents llibres relacionats amb l'enginyeria informàtica, com *Category Theory for Programmers* de Bartosz Milewski, *The Art of Computer Programming* de D. Knuth, *Clean Code* de Robert C. Martin [1, 2, 3]...

Un d'aquests llibres, *Algorithms to live by* [4] ha obert debats sobre diferents algorismes que desconeixíem i un d'aquests es va centrar en un tema concret del llibre que tractava el problema *explore-exploit* i d'una modelització d'aquest anomenat *k-armed bandits*. En el llibre, es comenta per sobre el que implica i quines possibles solucions té, així com també es parla d'una variació més complicada d'aquest problema, el *restless bandit*. La perspectiva que ens interessa d'aquest problema és el tipus de solució a la que s'arriba sabent la complexitat que té, que com es veurà, s'aconsegueix amb amb la relaxació del mateix.

2 Context

El teu estómac gruny. Vas al restaurant Italià que t'encanta, o proves el nou Tailandès que acaba d'obrir? Hi vas amb el teu millor amic o amb una persona que no coneixes tant i que vols conèixer millor? Massa difícil, millor quedar-se a casa. Cuines aquella recepta que t'encanta, o optes per buscar-ne alguna de nova per Internet? Saps que, potser millor demanar una pizza a domicili. Demanes la teva preferida o preguntes per les especials? Tant dubtar entre una o l'altra te n'hauràs cansat abans de poder fer la primera mossegada.

Cada dia ens veiem forçats a fer decisions entre dues opcions que difereixen en dues dimensions: Ens quedem amb les nostres coses preferides, o n'explorem de noves? Intuïtivament podem pensar que la vida és un balanç entre les dues, però la pregunta és: Quin és el balanç?

Molts matemàtics i informàtics han estat treballant en aquest balanç des de fa més de 50 anys donant-li el nom d'*explore/exploit tradeoff*.

En computació, la tensió entre explorar o explotar pren la seva forma més concreta en l'escenari anomenat *multi-armed bandit*, o *k-armed bandit*. Aquest nom li és donat per la forma col·loquial de referir-se a les màquines escura-butxaques. Imagina't entrar en un casino ple de màquines escura-butxaques, cada una amb les seves possibilitats de fer una tirada guanyadora. Naturalment, s'està interessat a maximitzar els guanys. Està clar que hi haurà una fase d'exploració en la qual testejarem les màquines, i una altra d'explotació tirant d'aquelles que creiem que són més beneficioses.

La primera passa cap a la solució va ser l'algorisme Win-Stay, Lose-Shift, proposat per Herbert Robins. Aquest consisteix a triar a una màquina aleatòria, mentre s'obtingui profit jugant en aquella màquina, es continua jugant en la mateixa i, si després d'una certa tirada la màquina deixa de ser profitosa, es canvia a una altra. Aquesta tot i estar lluny d'una solució optima, es va demostrar que els resultats eren millors que els de la pura sort.

No va ser fins al 1970 que John Gittins va trobar una solució optima que resolia el problema. Gittins va enfocar el problema en termes de maximitzar els guanys per un futur que és interminable però amb 'descomptes'. Fent així l'assumpció que el valor assignat als guanys decreixia geomètricament. Per exemple, es creu que hi ha un 1% de probabilitats de ser atropellat per un autobús un dia, aleshores s'ha de valorar el sopar del següent dia un 99% del valor del d'aquesta nit, només perquè l'endemà potser mai s'arriba a sopar. D'aquesta forma, va arribar a la conclusió que cada màquina de la qual en sabem una mica o res, té un nombre que ens indica la probabilitat de guany que ens farà decidir si tornar a jugar en ella o no. Aquest nombre és conegut com l'índex de Gittins.

Una variació d'aquest problema (*multi-armed bandit*), és que cada una de les

màquines escura-butxaques es comporta com una màquina Markov. És a dir, cada cop que una màquina en particular és jugada, l'estat d'aquesta canvia a un nou escollit d'acord a l'evolució de probabilitats dels estats d'aquesta màquina de Markov. I si a la variació anterior se li aplica, que l'estat de les màquines no jugades pot evolucionar al llarg del temps, apareix el problema del *restless bandit*.

3 Desenvolupament de la temàtica

Dins de les complexitats que poden tenir els problemes, la complexitat PSPACE compleix:

$$NP \subseteq PSPACE \subseteq EXP$$

De la mateixa manera que no es sap si $P = NP$ tampoc es sap si $NP = PSPACE$. La demostració que NP és contingut dins de PSPACE es realitza per reducció a l'absurd:

Sigui M una màquina de Turing NP, és a dir, donada una instància d'un problema ens diu en un temps polinomial si aquest pertany al problema. Si l'espai per a desenvolupar l'algorisme fos més gran a polinòmic, llavors forçosament per a llegir o escriure aquesta informació és necessari aquest temps, pel que arriba a l'absurd amb la definició de la màquina M.

El problema de *Restless bandit* es troba dins la complexitat PSPACE. Això va ser demostrat l'any 1999 en l'article [5].En aquest s'explica un problema de xarxes que és demostrar ser exponencial. Al mateix temps, un problema relaxat d'aquest es demostra ser PSPACE-complet, és a dir, tots els problemes de PSPACE poden ser reduïts a aquest problema i aquest pot ser reduït a tots els problemes de PSPACE. Finalment es dóna una fórmula de cost del problema de *restless bandit* i es redueix el problema relaxat a aquest, demostrant que és PSPACE-hard (tots els problemes de PSPACE són reduïbles a aquest).

3.1 Upper confidence bound

Que el problema es mostri com a PSPACE-hard significa que el càlcul del mínim és, si no es demostra la igualtat entre les diferents complexitats, tan dificultós que deixa de ser pràctic intentar-ho. Per sort, el problema relaxat a un *k-armed bandits* dona una bona heurística al problema principal. Aquest ja hem dit que és decidible amb una complexitat P per l'índex de Gittins, però el problema recau en el cost de calcular aquest índex, ja que encara que sigui polinomial en el temps el cost és bastant elevat, pel que s'utilitzen altres algoritmes que tot i no ser perfectes es queden molt propers a l'òptim del problema relaxat. Entre aquests algorismes trobem el d'*Upper confidence bound* (UCB), descrit en l'article [6].

L'algorisme UCB fa les seleccions de quina màquina jugar basant-se en l'optimisme. És a dir, centrant-se en el millor que podria tindre dur a terme una acció, donada la evidència obtinguda fins al moment. Seguint aquesta estratègia, una heurística possible és fer la selecció de la màquina seguint la següent formula:

$$A_n = argmax_a(Q_n(a) + c\sqrt{\frac{\log(n)}{k_n(a)}})$$

On $Q_n(a)$ és el valor mitjà actual d'obtenir una recompensa realitzant l'acció de jugar en una màquina escura-butxaques a. El valor sota l'arrel és el logaritme del nombre de màquines a les quals hem jugat, dividit per k_n, que és el nombre de tirades que hem fet en la màquina a. I finalment c, que és una constant a escollir.

Per calcular $Q_n(a)$ només es mantindran en memoria dos valors per cada acció, la mitja actual (m_n) i el nombre seleccions que s'han fet per arribar a aquesta acció k_n:

$$m_{n+1} = m_n + \frac{R_n - m_n}{k_n}$$

On R_n és la recompensa que obtenim de realitzar aquesta acció. Aleshores, cada cop que fem una jugada s'avaluaran totes les màquines amb les heurístiques que s'han definit i es jugarà en aquella que obtingui el valor màxim.

4 Conclusions

En aquest document s'ha explicat perquè el problema de *restless bandit*, encara que la seva relaxació sigui P, és PSPACE-hard. També s'ha pogut veure que a causa de la seva complexitat, un algorisme que retorni un resultat òptim és poc convenient, pel que és millor tenir un algorisme que no el resolgui òptimament sempre, si no un subòptim. Per a concluir el problema, s'ha ensenyat com funciona l'algorisme d'*Upper confidence bound*, que és un dels algorismes més utilitzats per a solucionar el problema. A més a més, també s'ha comentat amb anterioritat problemes que poden ser reduïts a aquest problema.

5 Bibliografia

[1] Bartosz Milewski. Category Theory for Programmers. Pàgina web, 2020.

[2] Donald Knuth. *The art of computer programming.* Addison-Wesley, Reading, Mass, 1997.

[3] Robert Martin. *Clean code : a handbook of agile software craftsmanship.* Prentice Hall, Upper Saddle River, NJ, 2009.

[4] Brian Christian. *Algorithms to live by : the computer science of human decisions.* Henry Holt and Company, New York, 2016.

[5] J. N. Tsitsiklis C. H. Papadimitriou. The complexity of optimal queueing network control. *Mathematics of Operations Research*, 24(2):293–305, May 1999.

[6] Christian Hubbs. Multi-Armed Bandits: UCB Algorithm. Article de medium, 2020.