

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	1
2	Context	1
3	Desenvolupament de la temàtica	3
3.1	Upper confidence bound	4
4	Conclusions	5

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 [?, ?, ?]...

Dins d'aquests llibres, *Algorithms to live by* [?] ha obert debats sobre diferents algorismes que desconexiem. Dins d'aquests, es comenta el problema de *k-armed bandits*, que implica i quina solució té. També es comenta superficialment *restless bandit*, que és una variació del problema més difícil. La perspectiva que ens interessa d'aquest és el tipus de solució que es dona sabent la complexitat que té, que és donar la relaxació del problema.

2 Context

El teu estomac gruny. Vas al restaurant Italià que t'encanta, o proves el nou Thaiandes que acaba d'obrir? Hi vas amb el teu millor amic o amb una persona que no coneixes tant i que vols conèixer millor? Masa 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 poguer fer la primera mossegada.

Cada dia ens veiem forçats a fer decisions entre dos opcions, que difereixen en dos dimensions: Ens quedem amb les nostres coses preferides, o n'explorem de noves? Intuitivament podem pensar que la vida és un balanç entre les dues, pero 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 de explore/exploit tradeoff.

En computació, la tensió entre explorar o explotar pren la seva forma més concreta an l'escenari anomenat multi-armed bandit, o k-armed bandit. Aquest nom li és donat degut a que es la forma coloquial de referir-se a les màquines escura-butxaques. Imagina entrar a un cassino ple de màquines escura-butxaques, cada una amb les seves possibilitats de fer una tirada guanyadora. Naturalment, s'està interessat en maximitzar els guanys. Està clar que hi haura una fase d'exploració en la qual testegarem 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 en triar a una màquina aleatoria, mentres 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 resolva el problema. Gittins va enfocar el problema en termes de maximitzar els guanys per un futur que és interminable pero amb 'descontes'. Fent així la assumsió de que el valor assignat als guanys decreixia geomètricament. Per exemple, es creu que hi ha un 1% de probabilitats de ser atropellat per un autobus un dia, aleshores s'ha de valorar el sopar del següent dia un 99% del valor del d'aquesta nit, només perque l'endemà potser mai s'arriba a sopar. D'aquesta forma, va arribar a la conclusió de 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'index 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. Es 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 de 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 es contingut dins de PSPACE es realitzar 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 es necessitaria 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 [?]. En aquest s'explica un problema de xarxes que es demostra 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 dona 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, és tan difícil 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 tot i que sigui polinomial en el temps el cost és bastant elevat, pel que s'utilitzen altres algorismes 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*.

L'algorisme UCB fa les seleccions de a quina màquina jugar en base a l'optimisme. Es 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:

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

On $Q_n(a)$ és el valor mitjà actual de 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 memòria 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'evaluaran 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, s'ha comentat amb anterioritat problemes que poden ser reduïts a aquest problema.