Linear Multi-Resource Allocation with Semi-Bandit Feedback

Authored by:

Koby Crammer
Csaba Szepesvari
Tor Lattimore

Abstract

We study an idealised sequential resource allocation problem. In each time step the learner chooses an allocation of several resource types between a number of tasks. Assigning more resources to a task increases the probability that it is completed. The problem is challenging because the alignment of the tasks to the resource types is unknown and the feedback is noisy. Our main contribution is the new setting and an algorithm with nearly-optimal regret analysis. Along the way we draw connections to the problem of minimising regret for stochastic linear bandits with heteroscedastic noise. We also present some new results for stochastic linear bandits on the hypercube that significantly out-performs existing work, especially in the sparse case.

1 Paper Body

Economist Thomas Sowell remarked that ?The first lesson of economics is scarcity: There is never enough of anything to fully satisfy all those who want it.?1 The optimal allocation of resources is an enduring problem in economics, operations research and daily life. The problem is challenging not only because you are compelled to make difficult trade-offs, but also because the (expected) outcome of a particular allocation may be unknown and the feedback noisy. We focus on an idealised resource allocation problem where the economist plays a repeated resource allocation game with multiple resource types and multiple tasks to which these resources can be assigned. Specifically, we consider a (nearly) linear model with D resources and K tasks. In each time step t the economist chooses an allocation of resources Mt ? RD?K where Mtk ? RD is the kth column and represents the amount of each resource type assigned to the kth task. We assume that the kth task is completed successfully with probability min {1, hMtk , ?k i} and ?k ? RD is an unknown non-negative vector that determines how the success rate of a given task depends on the quantity and type

of resources assigned to it. Naturally we will limit the availability of resources PK by demanding that Mt satisfies k=1 Mtdk ? 1 for all resource types d. At the end of each time step the economist observes which tasks were successful. The objective is to maximise the number of successful tasks up to some time horizon n that is known in advance. This model is a natural generalisation of the one used by Lattimore et al. [2014], where it was assumed that there was a single resource type only. 1 He went on to add that ?The first lesson of politics is to disregard the first lesson of economics.? Sowell [1993]

1

An example application might be the problem of allocating computing resources on a server between a number of Virtual Private Servers (VPS). In each time step (some fixed interval) the controller chooses how much memory/cpu/bandwidth to allocate to each VPS. A VPS is said to fail in a given round if it fails to respond to requests in a timely fashion. The requirements of each VPS are unknown in advance, but do not change greatly with time. The controller should learn which VPS benefit the most from which resource types and allocate accordingly. The main contribution of this paper besides the new setting is an algorithm designed for this problem along with theoretical guarantees on its performance in terms of the regret. Along the way we present some additional results for the related problem of minimising regret for stochastic linear bandits on the hypercube. We also prove new concentration results for weighted least squares estimation, which may be independently interesting. The generalisation of the work of Lattimore et al. [2014] to multiple resources turns out to be fairly non-trivial. Those with knowledge of the theory of stochastic linear bandits will recognise some similarity. In particular, once the nonlinearity of the objective is removed, the problem is equivalent to playing K linear bandits in parallel, but where the limited resources constrain the actions of the learner and correspondingly the returns for each task. Stochastic linear bandits have recently been generating a significant body of research (e.g., Auer [2003], Dani et al. [2008], Rusmevichientong and Tsitsiklis [2010], Abbasi-Yadkori et al. [2011, 2012], Agrawal and Goyal [2012] and many others). A related problem is that of online combinatorial optimisation. This has an extensive literature, but most results are only applicable for discrete action sets, are in the adversarial setting, and cannot exploit the additional structure of our problem. Nevertheless, we refer the interested reader to (say) the recent work by Kveton et al. [2014] and references there-in. Also worth mentioning is that the resource allocation problem at hand is quite different to the ?linear semi-bandit? proposed and analysed by Krishnamurthy et al. [2015] where the action set is also finite (the setting is different in many other ways besides). Given its similarity, it is tempting to apply the techniques of linear bandits to our problem. When doing so, two main difficulties arise. The first is that our payoffs are non-linear: the expected reward is a linear function only up to a point after which it is clipped. In the resource allocation problem this has a natural interpretation, which is that over-allocating resources beyond a certain point is fruitless. Fortunately, one can avoid this difficulty rather easily by ensuring that with high probability resources are never over-allocated. The second problem concerns achieving good

regret regardless of the task specifics. In particular, when the number of tasks K is large and resources are at a premium the allocation problem behaves more like a K-armed bandit where the economist must choose the few tasks that ? can be completed successfully. For this kind of problem regret should scale in the worst case with K only [Auer et al., 2002, Bubeck and Cesa-Bianchi, 2012]. The standard linear bandits approach, on the other hand, would lead to a bound on the regret that depends linearly on K. To remedy this situation, we will exploit that if K is large and resources are scarce, then many tasks will necessarily be under-resourced and will fail with high probability. Since the noise model is Bernoulli, the variance of the noise for these tasks is extremely low. By using weighted least-squares estimators we are able to exploit this and thereby obtain an improved regret. An added benefit is that when resources are plentiful, then all tasks will succeed with high probability under the optimal allocation, and in this case the variance is also low. This leads to a poly-logarithmic regret for the resource-laden case where the optimal allocation fully allocates every task.

2

Preliminaries

If F is some event, then ?F is its complement (i.e., it is the event that F does not occur). If A is positive definite and x is a vector, then kxk2A = x¿ Ax stands for the weighted 2-norm. We write —x— to be the vector of element-wise absolute values of x. We let ? ? RD?K be a matrix with columns ?1 , . . . ?K . All entries in ? are non-negative, but otherwise we make no global assumptions on ?. At each time step t the learner chooses an allocation matrix Mt ? M where ( ) K X D?K M = M ? [0, 1] : Mdk ? 1 for all d . k=1

The assumption that each resource type has a bound of 1 is non-restrictive, since the units of any resource can be changed to accommodate this assumption. We write Mtk ? [0, 1]D for the kth 2

column of Mt . The reward at time step t is kYt k1 where Ytk ? {0, 1} is sampled from a Bernoulli distribution with parameter ?(hMtk , ?k i) = min {1, hMtk , ?k i}. The economist observes all Ytk , however, not just the sum. The optimal allocation is denoted by M ? and defined by K X M ? = arg max ?(hMk , ?k i) . M ?M

k=1

We are primarily concerned with designing an allocation algorithm that minimises the expected (pseudo) regret of this problem, which is defined by " n K # K X XX ? Rn = n ?(hMk , ?k i) ? E ?(hMtk , ?k i) , t=1 k=1

k=1

where the expectation is taken over both the actions of the algorithm and the observed reward. Optimal Allocations If ? is known, then the optimal allocation can be computed by constructing an appropriate linear program. Somewhat surprisingly it may also be computed exactly in O(K log K + D log D) time using Algorithm 1 below. The optimal allocation is not so straightforward as, e.g., simply allocating resources to the incomplete task for which the corresponding ? is largest in some dimension. For example, for K = 2 tasks and d = 2 resource types:

0 1/2 0 1 ? = ?1 ?2 = =? M ? = M1? M2? = . 1/2 1 1/2 1/2 We see that

even though ?22 is the largest param- Algorithm 1 eter, the optimal allocation assigns only half of the Input: ? second resource (d = 2) to this task. The right apM = 0 ? RD?K and B = 1 ? RD proach is to allocate resources to incomplete tasks while ? k, d s.t hMk , ?k i ¡ 1 and Bd ¿ 0 do using the ratios as prescribed by Algorithm 1. The A = {k : hMk , ?k i ¡ 1} and B = {d : Bd ¿ 0}

intuition for allocating in this way is that resources ?dk k, d = arg max min should be allocated as efficiently as possible, and efi?A{k} ?di (k,d)?A?B ficiency is determined by the ratio of the expected

1 ? hMk , ?k i success due to the allocation of a resource and the Mdk = min Bd , ?dk amount of resources allocated. end while ? Theorem 1. Algorithm 1 returns M . return M The proof of Theorem 1 and an implementation of Algorithm 1 may be found in the supplementary material. We are interested primarily in the case when ? is unknown, so Algorithm 1 will not be directly applicable. Nevertheless, the algorithm is useful as a module in the implementation of a subsequent algorithm that estimates ? from data.

3

Optimistic Allocation Algorithm

We follow the optimism in the face of uncertainty principle. In each time step t, the algorithm constructs an estimator ??kt for each ?k and a corresponding confidence set Ctk for which ?k ? Ctk holds with high probability. The algorithm then takes the optimistic action subject to the assumption that ?k does indeed lie in Ctk for all k. The main difficulty is the construction of the confidence sets. Like other authors [Dani et al., 2008, Rusmevichientong and Tsitsiklis, 2010, Abbasi-Yadkori et al., 2011] we define our confidence sets to be ellipses, but the use of a weighted least-squares estimator means that our ellipses may be significantly smaller than the sets that would be available by using these previous works in a straightforward way. The algorithm accepts as input the number of tasks and resource types, the horizon and constants ? ¿ 0 and ? where constant ? is defined by D 1 2 , N = 4n4 D2 , B ? max k?k k2 , so that ?= k nK s

!2 ? 3nN 6nN ? = 1 + ?B + 2 log log . (1) ? ?

3

2

Note that B must be a known bound on maxk k?k k2 , which might seem like a serious restriction, until one realizes that it is easy to add an initialisation phase where estimates are quickly made while incurring minimal additional regret, as was also done by Lattimore et al. [2014]. The value of ? determines the level of regularisation in the least squares estimation and will be tuned later to optimise the regret. Algorithm 2 Optimistic Allocation Algorithm 1: Input K, D, n, ?, ? 2: for t ? 1, . . . , n do 3: // Compute confidence sets for all tasks k: P 4: Gtk = ?I + ? ¡t ?? k M? k M?¿k

6:

P ??tk = G?1 ? ¡t ?? k M? Y? k tk o o n n 2 2 0 = ??k : k? ?k ? ??tk kGtk ? 4? Ctk = ??k : k? ?k ? ??tk kGtk ? ? and Ctk

7: 8:

// Compute optimistic allocation: Mt = arg maxMt ?M max??k ?Ctk ?(hMtk , ??k i)

5:

9: 10:

// Observe success indicators Ytk for all tasks k: Ytk ? Bernoulli(?(hMtk , ?k i))

11: // Compute weights for all tasks k: ?1 12: ?tk = arg max??k ?Ctk ?k i (1 ? hMtk , ??k i) 0 hMtk , ? 13: end for

Computational Efficiency We could not find an efficient implementation of Algorithm 2 because solving the bilinear optimisation problem in Line 8 is likely to be NP-hard (Bennett and Mangasarian [1993] and also Petrik and Zilberstein [2011]). In our experiments we used a simple algorithm based on optimising for M and ? in alternative steps combined with random restarts, but for large D and K this would likely not be efficient. In the supplementary material we present an alternative algorithm that is efficient, but relies on the assumption that k?k k1 ? 1 for all k. In this regime it is impossible to over-allocate resources and this fact can be exploited to obtain an efficient and practical algorithm with strong guarantees. Along the way, we are able to construct an elegant algorithm for linear bandits on the hypercube that enjoys optimal regret and adapts to sparsity. Computing the weights ?tk (Line 12) is (somewhat surprisingly) straight-forward. Define p p and ptk = hMtk , ??tk i ? 2 ? kMtk kG?1 . p?tk = hMtk , ??tk i + 2 ? kMtk kG?1 tk

tk

Then the weights can be computed by ?1 ?tk

? 1 ?p?tk (1 ? p?tk) if p?tk ? 2 = ptk (1 ? ptk) if ptk ? 21 ?1 otherwise . 4

(2)

A curious reader might wonder why the weights are computed by optimising within confidence set 0 Ctk , which has double the radius of Ctk . The reason is rather technical, but essentially if the true parameter ?k were to lie on the boundary of the confidence set, then the corresponding weight could become infinite. For the analysis to work we rely on controlling the size of the weights. It is not clear whether or not this trick is really necessary.

4

Worst-case Regret for Algorithm 2

We now analyse the regret of Algorithm ? 2. First we offer a worst-case bound on the regret that depends on the time-horizon like O(n). We then turn our attention to the resource-laden case where the optimal allocation satisfies hMk? , ?k i = 1 for all k. In this instance we show that the dependence on the horizon is only poly-logarithmic, which would normally be unexpected when the 4

action-space is continuous. The improvement comes from the weighted estimation that exploits the fact that the variance of the noise under the optimal allocation vanishes. 2

Theorem 2. Suppose Algorithm 2 is run with bound B ? maxk k?k k2 . Then s

p Rn ? 1 + 4D 2?nK max k?k k? + 4 ?/? log(1 + 4n2) . k
Choosing ? = B ?1 log
6nN ?
3nN ?
2
and assuming that B ? O(maxk k?k k2), then
q 3/2 Rn ? O D nK max k?k k2 log n . log
k

The proof of Theorem 2 will follow by carefully analysing the width of the confidence sets as the algorithm makes allocations. We start by proving the validity of the confidence sets, and then prove the theorem. Weighted Least Squares Estimation For this sub-section we focus on the problem of estimating a single unknown ? = ?k . Let n M1 , . . . , Mn be a sequence of allocations to task k with Mt ? RD . Let {Ft }t=0 be a filtration with Ft containing information available at the end of round t, which means that Mt is Ft?1 measurable. Let ?1 , . . . , ?n be the sequence of weights chosen by Algorithm 2. The sequence of outcomes is Y1 , . . . , Yn ? P {0, 1} for which E[Yt —Ft?1] = ?(hMt , ?i). The weighted regularised gram matrix is Gt = ?I + ? ¡t ?? M? M?¿ and the corresponding weighted least squares estimator is X ??t = G?1 ?t M? Y? . t ? ¡t 2
2

Theorem 3. If k?k2 ? B and ? is chosen as in Eq. (1), then k? ?t ? ?kGt ? ? for all t ? n with probability at least 1 ? ? = 1/(nK). Similar results exist in the literature for unweighted least-squares estimators (for example, Dani et al. [2008], Rusmevichientong and Tsitsiklis [2010], Abbasi-Yadkori et al. [2011]). In our case, however, Gt is the weighted gram matrix, which may be significantly larger than an unweighted version when the weights become large. The proof of Theorem 3 is unfortunately too long to include in the main text, but it may be found in the supplementary material. Analysing the Regret 2

We start with some technical lemmas. Let F be the failure event that k? ?tk ? ?k kGtk ¿ ? for some t ? n and 1 ? k ? K. Lemma 4 (Abbasi-Yadkori et al. [2012]). Let x1 , . . . , xn be n an arbitrary o sequence of vectors with Pt?1 Pn 2 ¿ 2 kxt k2 ? c and let Gt = I + s=1 xs xs . Then t=1 min 1, kxt kG?1 ? 2D log 1 + c?n D . t

Corollary 5. If F does not hold, then
n X
n o 2 ?tk min 1, kMtk kG?1 ? 8D log(1 + 4n2). tk
t=1

The proof is omitted, but follows rather easily by showing that ?tk can be moved inside the minimum at a price of increasing the loss at most by a factor of four, and then applying Lemma 4. See the supplementary material for the formal proof.

K X p ?1 Lemma 6. Suppose F does not hold, then ?tk ? D max k?k k? + 4 ?/? . k
k=1
5

?1 Proof. We exploit the fact that ?tk is an estimate of the variance, which is small whenever kMtk k1 is small: ?1 ?tk = arg max hMtk , ??k i (1 ? hMtk , ??k i) ? arg max hMtk , ??k i 0 ? ?k ?Ctk

0 ? ?k ?Ctk

(a)

= hMtk , ?i + arg max hMtk , ??k ? ?i ? kMtk k1 k?k k? + 4

p

? ?k ?Ctk0

(b)

? kMtk k1 k?k k? + 4

p

? kMtk kG?1 tk

(c) p ? kMtk kI/? ? kMtk k1 k?k k? + 4 ?/? ,

?1 0 where (a) follows from Cauchy-Schwartz k ? Ctk , (b) since Gtk ? I/? and basic pand the fact that ?p linear algebra, (c) since kMtk kI/? = 1/? kMtk k2 ? 1/? kMtk k1 . The result is completed PK since the resource constraints implies that k=1 kMtk k1 ? D.

Proof of Theorem 2. By Theorem 3 we have that F holds with probability at most ? = 1/(nK). If F does not hold, then by the definition of the confidence set we have ?k ? Ctk for all t and k. Therefore " # n X K n X K X X Rn = E (hMk? , ?k i ? ?(hMtk , ?k i)) ? 1 + E 1 {?F } hMk? ? Mtk , ?k i . t=1 k=1

t=1 k=1

Note that we were able to replace ?(hMtk , ?k i) = hMtk , ?k i, since if F does not hold, then Mtk will never be chosen in such a way that resources are over-allocated. We will now assume that F does not hold and bound the argument in the expectation. By the optimism principle we have: n X K X

hMk?

(a)

? Mtk , ?k i ?

t=1 k=1

n X K X

min {1, hMtk , ??tk ? ?k i}

t=1 k=1 (b)

?

K n X X

o n ?tk ? ?k kGtk min 1, kMtk kG?1 k? tk

t=1 k=1 (c)

?2

n X K X

n p o min 1, kMtk kG?1 ? tk

t=1 k=1

v u n u X ? ? 2tn

(d)

K X

! n o 2 min 1, kMtk kG?1 tk

t=1

v u n u X ? 2tn ?
(e)
t=1
k=1 K X
! ?1 ?tk
k=1
K X
! n o 2 ?tk min 1, kMtk kG?1 tk
k=1

v ! r ! n u K n o u X ? X 2 t ? 2 nD max k?k k? + 4 ? ?tk min 1, kMtk kG?1 tk k ? t=1 k=1 v r ! u u (g) ? t ? 4D 2?nK max k?k k? + 4 log(1 + 4n2) . k ?
(f)

where (a) follows from the assumption that ?k ? Ctk for all t and k and since Mt is chosen optimistically, (b) by the Cauchy-Schwarz inequality, (c) by the definition of ??kt , which lies inside Ctk , (d) by Jensen?s inequality, (e) by Cauchy-Schwarz again, (f) follows from Lemma 6. Finally (g) follows from Corollary 5.

5

Regret in Resource-Laden Case

We now show that if there are enough resources such that the optimal strategy can complete ? every task with certainty, then the regret of Algorithm 2 is poly-logarithmic (in contrast to O(n) otherwise). As before we exploit the low variance, but now the variance is small because hMtk , ?k i is 6

close to 1, while in the previous section we argued that this could not happen too often (there is no contradiction as the quantity maxk k?k k appeared in the previous bound). PK Theorem 7. If k=1 hMk? , ?k i = K, then Rn ? 1 + 8?KD log(1 + 4n2). Proof. We start by showing that the weights are large: ?1 ?tk = max0 hMtk , ?i (1 ? hMtk , ?i) ? max0 (1 ? hMtk , ?i) ??Ctk
??Ctk
p ? max0 hMtk , ?? ? ?i ? kMtk kG?1 max0 k? ? ? ?kGtk ? kMtk kG?1 4 ? . ? ?,??Ctk
tk
? ?,??Ctk
tk

Applying the optimism principle and using the bound above combined with Corollary 5 gives the result: " # n X K X ERn ? 1 + E 1 {?F } min {1, hMtk , ??kt ? ?k i} t=1 k=1
" ? 1 + 2E 1 {?F }
n X K X
# n p o min 1, kMtk kG?1 ? tk
t=1 k=1
" = 1 + 2E 1 {?F }
K n X X
n op ?1 ? min 1, ?tk ?tk kMtk kG?1
#

tk
t=1 k=1
” ? 1 + 8? E 1 {?F }
K n X X
# n o 2 min 1, ?tk kMtk kG?1 tk
t=1 k=1 2
? 1 + 8?KD log(1 + 4n) .
6
Experiments

We present two experiments to demonstrate the behaviour of Algorithm 2. All code and data is available in the supplementary material. Error bars indicate 95% confidence intervals, but sometimes they are too small to see (the algorithm is quite conservative, so the variance is very low). We used B = 10 for all experiments. The first experiment demonstrates the improvements obtained by using a weighted estimator over an unweighted one, and also serves to give some idea of the rate of learning. For this experiment we used D = K = 2 and n = 106 and

K X 8/10 2/10 1 0 ? ? = ?1 ?2 = hMk? , ?k i = 2 , =? M = and 4/10 2 1/2 1/2 k=1

where the kth column is the parameter/allocation for the kth task. We ran two versions of the algorithm. The first, exactly as given in Algorithm 2 and the second identical except that the weights were fixed to ?tk = 4 for all t and k (this value is chosen because it corresponds to the minimum inverse variance for a Bernoulli variable). The data was produced by taking the average regret over 8 runs. The results are given in Fig. 1. In Fig. 2 we plot ?tk . The results show that ?tk is increasing linearly with t. This is congruent with what we might expect because in this regime the estimation error should drop with O(1/t) and the estimated variance is proportional p to the estimation error. Note that the estimation error for the algorithm with ?tk = 4 will be O(1/t). For the second experiment we show the algorithm adapting to the environment. We fix n = 5 ? 105 and D = K = 2. For ? ? (0, 1) we define

K X 1/2 ?/2 1 0 and hMk? , ?k i = 1 . ?? = =? M? = 1/2 ?/2 1 0 k=1

The unusual profile of the regret as ? varies can be attributed to two factors. First, if ? is small then the algorithm quickly identifies that resources should be allocated first to the first task. However, in the early stages of learning the algorithm is conservative in allocating to the first task to avoid overallocation. Since the remaining resources are given to the second task, the regret is larger for small 7

? because the gain from allocating to the second task is small. On the other hand, if ? is close to 1, then the algorithm suffers the opposite problem. Namely, it cannot identify which task the resources should be assigned to. Of course, if ? = 1, then the algorithm must simply learn that all resources can be allocated safely and so the regret is smallest here. An important point is that the algorithm never allocates all its resources at the start of the process because this risks over-allocation, so even in ?easy? problems the regret will not vanish.
Figure 1: Weighted vs unweighted estimation

Figure 3: ?Gap? dependence
Figure 2: Weights
80,000
30,000
40
20,000
20
Weighted Estimator Unweighted Estimator
0 0
1,000,000 t
7
Regret
40,000
?
Regret
60,000
?t1 ?t2
0 0
1,000,000 t
20,000 10,000 0 0.0
0.5 ?
1.0
Conclusions and Summary

We introduced the stochastic multi-resource allocation problem and developed a new algorithm that enjoys near-optimal worst-case regret. The main drawback of the new algorithm is that its computation time is exponential in the dimension parameters, which makes practical implementations challenging unless both K and D are relatively small. Despite this challenge we were able to implement that algorithm using a relatively brutish approach to solving the optimisation problem, and this was sufficient to present experimental results on synthetic data showing that the algorithm is behaving as the theory predicts, and that the use of the weighted least-squares estimation is leading to a real improvement. Despite the computational issues, we think this is a reasonable first step towards a more practical algorithm as well as a solid theoretical understanding of the structure of the problem. As a consolation (and on their own merits) we include some other results: ? An efficient (both in terms of regret and computation) algorithm for the case where overallocation is impossible. ? An algorithm for linear bandits on the hypercube that enjoys optimal regret bounds and adapts to sparsity. ? Theoretical analysis of weighted least-squares estimators, which may have other applications (e.g., linear bandits with heteroscedastic noise). There are many directions for future research. The most natural is to improve the practicality of the algorithm. We envisage such an algorithm might be obtained by following the program below: ? Generalise the Thompson sampling analysis for linear bandits by Agrawal and Goyal [2012]. This is a highly non-trivial step, since it is no longer straight-forward to show that such an algorithm is optimistic with high probability. Instead it will be

necessary to make do with some kind of local optimism for each task. ? The method of estimation depends heavily on the algorithm over-allocating its resources only with extremely low probability, but this significantly slows learning in the initial phases when the confidence sets are large and the algorithm is acting conservatively. Ideally we would use a method of estimation that depended on the real structure of the problem, but existing techniques that might lead to theoretical guarantees (e.g., empirical process theory) do not seem promising if small constants are expected. It is not hard to think up extensions or modifications to the setting. For example, it would be interesting to look at an adversarial setting (even defining it is not so easy), or move towards a nonparametric model for the likelihood of success given an allocation. 8

2 References

Yasin Abbasi-Yadkori, Csaba Szepesv?ari, and David Tax. Improved algorithms for linear stochastic bandits. In Advances in Neural Information Processing Systems, pages 2312?2320, 2011. Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Online-to-confidence-set conversions and application to sparse stochastic bandits. In AISTATS, volume 22, pages 1?9, 2012. Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. arXiv preprint arXiv:1209.3352, 2012. Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. The Journal of Machine Learning Research, 3:397?422, 2003. Peter Auer, Nicol?o Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. Machine Learning, 47:235?256, 2002. Kristin P Bennett and Olvi L Mangasarian. Bilinear separation of two sets inn-space. Computational Optimization and Applications, 2(3):207?227, 1993. S?ebastien Bubeck and Nicol'o Cesa-Bianchi. Regret Analysis of Stochastic and Nonstochastic Multiarmed Bandit Problems. Foundations and Trends in Machine Learning. Now Publishers Incorporated, 2012. ISBN 9781601986269. Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In COLT, pages 355?366, 2008. Akshay Krishnamurthy, Alekh Agarwal, and Miroslav Dudik. Efficient contextual semi-bandit learning. arXiv preprint arXiv:1502.05890, 2015. Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvari. Tight regret bounds for stochastic combinatorial semi-bandits. arXiv preprint arXiv:1410.0949, 2014. Tor Lattimore, Koby Crammer, and Csaba Szepesv?ari. Optimal resource allocation with semi-bandit feedback. In Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI), 2014. Marek Petrik and Shlomo Zilberstein. Robust approximate bilinear programming for value function approximation. The Journal of Machine Learning Research, 12:3027?3063, 2011. Paat Rusmevichientong and John N Tsitsiklis. Linearly parameterized bandits. Mathematics of Operations Research, 35(2):395?411, 2010. Thomas Sowell. Is Reality Optional?: And Other Essays. Hoover Institution Press, 1993.

9