

## Exercise 2

a)

Clearly the discrepancy is at least one, as  $|\chi(\{1\})| = 1$ .

Chose  $\chi(i) = (-1)^i$  for all  $1 \leq i \leq n$ . This gives  $|\chi(\{i\})| = 1$  for all  $1 \leq i \leq n$  and  $|\chi([n])| = 0$  if  $n$  is even and  $|\chi([n])| = 1$  if  $n$  is odd. Hence the discrepancy is at most one.

b)

The determinant of this matrix is 2 and hence the matrix is not totally unimodular.

c)

Let  $L = \lceil n/2 \rceil$  and  $R = [n] \setminus L$ . Consider  $y$  defined as  $y_i = 1$  for  $i \in L$  and  $y_i = 0$  otherwise. Let  $S \subseteq [n]$ . Then  $|A_S y| \leq \max(1/2|\chi(S \cap L)|, 1/2|\chi(S \cap R)|) \leq 1/2\lceil n/2 \rceil$ . This shows that the discrepancy is at most  $1/2\lceil n/2 \rceil$ .

Consider any  $y$ . Let  $L' = \{i \in [n] \mid y_i = 1\}$  and  $R' = [n] \setminus L'$ . Then  $|A_{L'} y| = |L'|/2$  and  $|A_{R'} y| = |R'|/2$ . As one of the sets has cardinality at least  $\lceil n/2 \rceil$  the claim follows.

d)

Approach 1:

From the lecture, it is known that the discrepancy of a totally unimodular matrix is less than one. From the definition of the discrepancy it is obvious that the discrepancy is integral if  $Ax$  is integral. Hence the discrepancy must be 0 in this case.

Approach 2:

Let  $b = Ax$ . Consider  $P = \{Ay = b, y \geq 0, y \leq 1\}$ . As  $A$  is totally unimodular,  $\begin{pmatrix} A \\ I \\ -I \end{pmatrix}$  is totally unimodular. As  $P$  is non-empty (as  $x \in P$ ) and bounded, it contains a vertex. As any vertex of  $P$  is integral, any vertex of  $P$  gives a  $y$  with  $Ay = Ax$ . Hence the discrepancy is 0.

## Exercise 4

a)

$$\min \sum_i x_i \tag{1}$$

$$Ax^T \geq 1 \tag{2}$$

$$x \in \{0, 1\} \tag{3}$$

decision variable  $x_i$  is 1 iff  $v_i \in V_0$

$A$  is the edge-vertex incidence matrix for hypergraphs

(rows correspond to edges and contain a 1 in all columns corresponding to connected vertices)

(1) clearly minimizes the number of vertices in  $V_0$

(2) for each row (edge), the number of vertices in  $V_0$  which are hit has to be at least 1

(3) ILP

b)

$$\min z \tag{4}$$

$$z \geq (A(x - y))_i \forall i \tag{5}$$

$$z \geq (A(y - x))_i \forall i \tag{6}$$

$$y \in \{0, 1\} \tag{7}$$

$$\text{mindisc}(A, x) = \min_{y \in \{0, 1\}} \|A(x - y)\|_\infty = \min_{y \in \{0, 1\}} \max_i |A(x - y)_i|$$

the absolute value can be rewritten as constraint (5) and (6) (notice switched signs) and having two constraints for each row in the result vector  $A(x - y)$ .

(4) together with (5) and (6) reduces the min-max problem into a regular min problem

(7) ILP

c)

$$\max \sum_{i,j} e_{ij} \tag{8}$$

$$e_{ij} \leq v_i + v_j \forall (i, j) \in E \tag{9}$$

$$v_i + e_{ij} + v_j \leq 2 \forall (i, j) \in E \tag{10}$$

$$v_i \in 0, 1 \tag{11}$$

$$e_{ij} \in 0, 1 \tag{12}$$

decision variable  $e_{ij}$  is 1 iff  $e_{ij} = (i, j) \in C$

decision variable  $v_i$  is 1 iff  $v_i \in U$

(8) clearly maximizes the cardinality of the cut

(9) an edge  $e_{ij}$  can only be  $\in C$  iff  $v_i \in U$  or  $v_j \in U$

(10) but an edge  $e_{ij}$  can only be  $\in C$  iff not both  $v_i \in U$  and  $v_j \in U$

(11), (12) ILP

## Exercise 5

The key problem was generating a suitable random bisector. If you simply assign each vertex to either  $U$  or  $V \setminus U$ , each with probability  $\frac{1}{2}$ , then with high probability  $|U| \neq \frac{1}{2}|V|$ . Such solutions were granted zero points, mainly because this was simply repeating the corresponding result on MaxCut from the lecture.

The *natural solution* is taking a random  $U \subseteq V$  with  $|U| = \frac{1}{2}|V|$ , where random shall mean that each such  $U$  has the same probability. This raises the problem of how to actually generate such a random  $U$ . Let us for simplicity assume that  $V = [n]$  for some even integer  $n$ . Then the following does not suffice: For  $v = 1 \dots n$  put  $v$  in  $U$  or  $V \setminus U$  with probability  $\frac{1}{2}$  until one of the two sets has  $n/2$  elements (“is full”). Then put the rest in the other set. Obviously, here it is very likely that  $n-1$  and  $n$  go to the same set. Hence, this  $U$  is not random in the sense of above.

A random  $U$  can e.g. be generated as follows: Put  $U := \emptyset$ . For  $i = 1 \dots n/2$  choose a random vertex from  $V \setminus U$  and put it to  $U$ .

The second thing to note is that for such a random  $U$  the events  $x \in U$  and  $y \in V \setminus U$  for different  $x, y \in V$  are not independent: Once we know that  $x \in U$ , then  $y$  has a slightly higher chance to be in  $V \setminus U$ , namely  $\frac{1}{2}n/(n-1)$ . Noting this, we can employ the same proof method as in the lecture. Since  $\frac{1}{2}n/(n-1) > \frac{1}{2}$ , we actually get a slightly better result (not asked for in the exam).

The *clever solution* is this: For each  $i = 1 \dots n/2$  choose a random bit  $b_i \in \{0, 1\}$ . If  $b_i = 0$ , put  $U := U \cup \{2i-1\}$ , else put  $U := U \cup \{2i\}$ . In other words, for each set  $\{2i-1, 2i\}$  we randomly put exactly one of its elements into  $U$ . Besides being a simple algorithm, this is also easy to analyse. For an edge  $e = \{u, v\} \in E$ , there are two possibilities. Either  $e = \{2i-1, 2i\}$  for some  $i$ . Then  $e$  is in the cut with probability one. Otherwise, the events  $u \in U$  and  $v \in V \setminus U$  are independent. Then  $e$  is in the cut with probability  $\frac{1}{2}$  as in the proof in the lecture. Congratulations to those who found this neat idea!

## Optimization – Final Exam

### Solutions:

#### 1. Multiple choice

(a) FALSE 1p

(b) TRUE 1p

(c) FALSE 1p

(d) FALSE 1p

(e) TRUE 1p

(f) TRUE 1p

(g) TRUE 1p

(h) FALSE 1p

(i) FALSE 1p

## Optimization – Final Exam

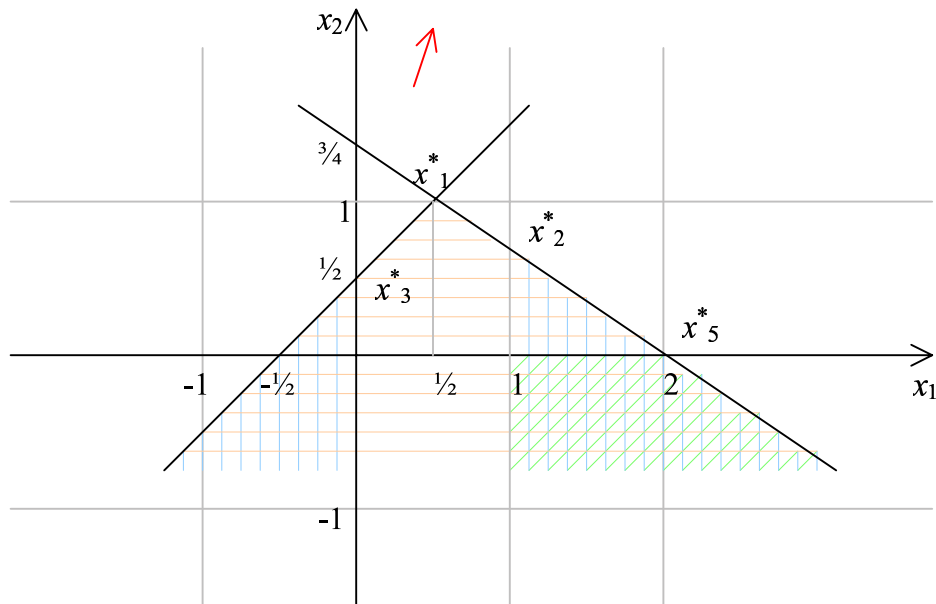
### Solutions

#### 3. Branch & Bound

Linear relaxation of the ILP:

$$\begin{aligned} \max \quad & \frac{1}{2}x_1 + 2x_2 && (P_1) \\ \text{s.t.} \quad & \frac{1}{2}x_1 + \frac{3}{4}x_2 \leq 1 \\ & -2x_1 + 2x_2 \leq 1 \\ & x_1 \in \mathbf{Z} \end{aligned}$$

Feasible region (figure):



Solution of the relaxation  $P_1$ :  $x_1^* = (\frac{1}{2}, 1)$ , the objective value:  $z = 9/4$

Branching with respect to  $x_1$ , which is not integer:

- add the constraint  $x_1 \geq \lceil \frac{1}{2} \rceil = 1$  and call the new LP ( $P_2$ )
- add the constraint  $x_1 \leq \lfloor \frac{1}{2} \rfloor = 0$  and call the new LP ( $P_3$ )

Solution of  $P_2$ :  $x_2^* = (1, 2/3)$ , the objective value:  $z = 11/6$ .

Solution of  $P_3$ :  $x_3^* = (0, 1/2)$ , the objective value:  $z = 1$ .

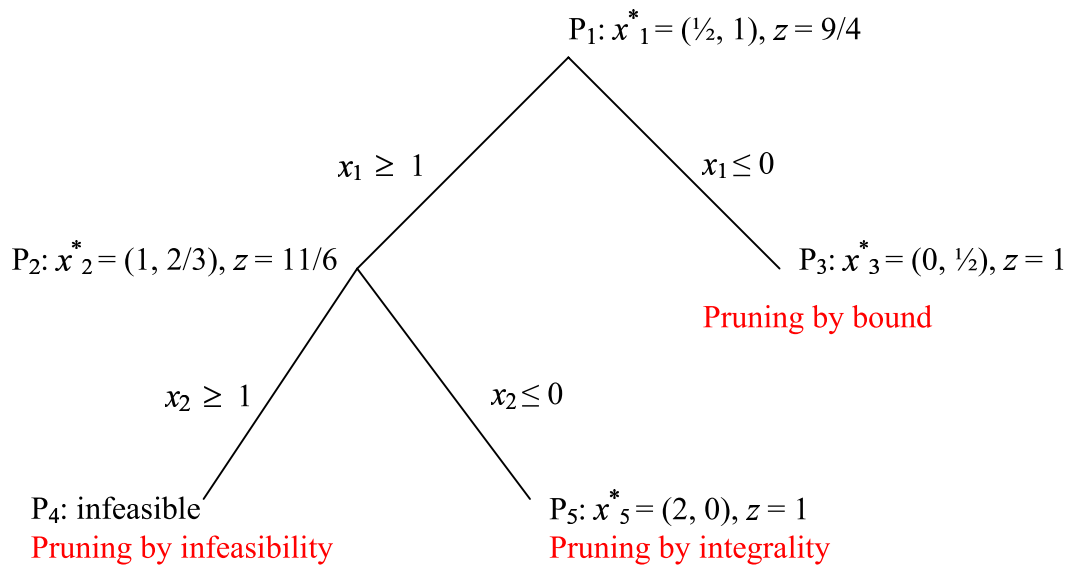
Branching for  $P_2$  with respect to  $x_2$ , which is not integer:

- add the constraint  $x_2 \geq \lceil 2/3 \rceil = 1$  and call the new LP ( $P_4$ )
- add the constraint  $x_2 \leq \lfloor 2/3 \rfloor = 0$  and call the new LP ( $P_5$ )

$P_4$  is infeasible. We prune the Branch and Bound tree by infeasibility.

Solution for  $P_5$ :  $x_5^* = (2, 0)$ , the objective value:  $z = 1$ . Since  $x_5^*$  is integer, we prune the tree by integrality.

There is no need to further branch  $P_3$ , as it has objective value 1 and it can only get worse. We prune the tree by bound and the ILP is solved, with optimal integer solution  $x_5^* = (2, 0)$  and objective function value  $z = 1$ .



Feasible region of the relaxation: correct figure – 1p

Solution of the relaxation – 1p

Correctly using the Branch and Bound method – 2p

Correct optimum solution for the ILP – 1p

### Sample Solution for Problem 6

Consider the following algorithm which computes a greedy packing of  $n$  items with weights  $w : [n] \rightarrow [0, 1]$ .

1. Set  $k := 0$  initially.
2. For each item  $i \in [n]$  in increasing order do the following
  - (a) If there is an index  $j \in [k]$  with  $w(B_j) + w(i) \leq 1$  then update the  $j$ -th bin

$$B_j := B_j \cup \{i\}$$

- (b) otherwise open a new bin

$$B_{k+1} := \{i\}$$

and update the number of bins  $k := k + 1$ .

3. Return the packing  $B_1 \dot{\cup} \dots \dot{\cup} B_k = [n]$ .

The running time is  $\mathcal{O}(n \cdot k)$  and thus polynomial in the input size. The algorithm is correct since each item is packed into exactly one bin and

$$\forall j \in [k]. w(B_j) \leq 1$$

is an invariant of the algorithm.

Let  $B_1 \dot{\cup} \dots \dot{\cup} B_k$  be a greedy packing of  $n$  items. We prove by induction on  $n \geq 1$  the following inequality.

$$|\{j \in [k] \mid 2w(B_j) \geq 1\}| \geq k - 1 \tag{1}$$

For  $n = 1$ , we have  $k = 1$  and therefore (1) is true.

Assume  $n > 1$ . Let bin index  $j_0 \in [k]$  be such that  $n \in B_{j_0}$ . If  $B_{j_0}$  contains more than one item then

$$B'_1 \dot{\cup} \dots \dot{\cup} B'_k \text{ where } B'_j := \begin{cases} B_j \setminus \{n\} & j = j_0 \\ B_j & \text{otherwise} \end{cases}$$

is a greedy packing of  $n - 1$  items. We have

$$k - 1 \leq |\{j \in [k] \mid 2w(B'_j) \geq 1\}| \leq |\{j \in [k] \mid 2w(B_j) \geq 1\}|$$

where the first inequality is by induction hypothesis and the second is by construction. Hence (1) holds also in this case.

Assume  $B_{j_0} = \{n\}$ . By construction of the algorithm, we have  $j_0 = k$ . If  $2w(n) \geq 1$  then

$$\begin{aligned} |\{j \in [k] \mid 2w(B_j) \geq 1\}| &= 1 + |\{j \in [k-1] \mid 2w(B_j) \geq 1\}| \\ &\geq 1 + (k-1) - 1 \\ &= k - 1 \end{aligned} \tag{*}$$

where (\*) is by induction hypothesis on  $B_1 \dot{\cup} \dots \dot{\cup} B_{k-1}$  which happens to be a greedy packing of  $n - 1$  items.

If  $2w(n) < 1$  then by construction of the algorithm,  $2w(B_j) \geq 1$  holds for all  $j \in [k-1]$  and therefore

$$|\{j \in [k] \mid 2w(B_j) \geq 1\}| = |\{1, \dots, k-1\}| = k - 1.$$

Hence (1) is true in all cases. □

It remains to show that  $k \leq 2OPT + 1$  for any greedy packing  $B_1 \dot{\cup} \dots \dot{\cup} B_k$ .

*Proof.* Using (1) we get

$$k = 1 + k - 1 \leq 1 + \sum_{j \in [k]} 2w(B_j) = 1 + \sum_{i \in [n]} 2w(i) \leq 1 + 2OPT$$

where the last inequality is true since  $w([n]) \leq OPT$  as each bin can hold weight at most 1. □