Question 1.

For $k \in \mathcal{K}, n \in \mathcal{N}, m = 1, \ldots, M$, we define the binary variables $x_{k,m,n} \in \{0,1\}$ such that $x_{k,m,n} = 1$ if and only if $p_{k,m,n} \leq p_{k,n} < p_{k,m+1,n}$, with $p_{k,M+1,n}$ interpreted as $+\infty$. Thus, the constraint by the total power budget is

$$\sum_{\substack{k\in\mathcal{K}\\ n\in\mathcal{N}\\ m=1,\ldots,M}} p_{k,m,n}x_{k,m,n} \leq p$$

the constraint that each channel serves one and only one user is

$$\sum_{\substack{k\in\mathcal{K}\\ m=1,\ldots,M}} x_{k,m,n} = 1,\ \forall n \in \mathcal{N}$$

and the target function is

$$U := \sum_{\substack{k\in\mathcal{K}\\ n\in\mathcal{N}\\ m=1,\ldots,M}} r_{k,m,n}x_{k,m,n}$$

In all, we have the ILP below

$$\begin{aligned} x_{k,m,n} &\in \{0,1\} && (1)\\ \sum_{k,m,n} p_{k,m,n}x_{k,m,n} &\leq p && (2)\\ \sum_{k,m} x_{k,m,n} &= 1,\ \forall n \in \mathcal{N} && (3) \end{aligned}$$

with target function

$$U := \sum_{k,m,n} r_{k,m,n}x_{k,m,n}$$

the corresponding LP is obtained by replacing (1) with $x_{k,m,n} \in [0,1]$

Question 2.

Proof of **Lemma 1**

Suppose $p_{k,m,n} \leq p_{k',m',n}$ and $r_{k,m,n} \geq r_{k',m',n}$. Given an optimal solution to the ILP, if $x_{k',m',n} \neq 0$, then $x_{k',m',n} = 1$ by (1). If we replace $x_{k,m,n}$ by 1 and $x_{k',m',n}$ by 0, (1) and (3) are clearly satisfied, (2) is also satisfied since $p_{k,m,n} \leq p_{k',m',n}$, and U will not decrease since $r_{k,m,n} \geq r_{k',m',n}$, so we get an optimal where $x_{k',m',n} = 0$

Question 3.

REMOVE-IP-DOMINATED(n)

1 sort the pairs $(p_{k,m,n}, r_{k,m,n})$ in increasing order of $p_{k,m,n}$ into an array A
2 cm = A[0].r
3 **for** i = 0 to A.length-1
4   **if** A[i].r $\geq$ cm
5   cm = A[i].r
6   **else** remove A[i].p and A[i].r from the original data

sorting $p_{k,m,m}$ takes time $O(KM\log(KM))$, the loop from line 3 takes time $O(KM)$. We will run REMOVE-IP-DOMINATED for each $n \in \mathcal{N}$, so in all it takes time $O(NKM\log(KM))$ to remove IP-dominated terms.
N.B. Accounting for that $p_{k,m,n}$ is increasing according to m for k, n fixed, we may sort quicker in line 1 and achieve a complexity of $O(NKM\log(K)$.

Question 4.

Proof of **Lemma 2**

Suppose $p_{k,m,n} \leq p_{k',m',n} \leq p_{k'',m'',n}$ and

$$\frac{r_{k'',m'',n} - r_{k',m',n}}{p_{k'',m'',n} - p_{k',m',n}} \geq \frac{r_{k',m',n} - r_{k,m,n}}{p_{k',m',n} - p_{k,m,n}} \tag{4}$$

note that we have

$$p_{k',m',n} = p_{k,m,n}\frac{p_{k'',m'',n} - p_{k',m',n}}{p_{k'',m'',n} - p_{k,m,n}} + p_{k'',m'',n}\frac{p_{k',m',n} - p_{k,m,n}}{p_{k'',m'',n} - p_{k,m,n}} \tag{5}$$

and from (4) we can deduce that

$$r_{k',m',n} \leq r_{k,m,n}\frac{p_{k'',m'',n} - p_{k',m',n}}{p_{k'',m'',n} - p_{k,m,n}} + r_{k'',m'',n}\frac{p_{k',m',n} - p_{k,m,n}}{p_{k'',m'',n} - p_{k,m,n}} \tag{6}$$

Given an optimal solution to the LP, we can construct another solution with

$$\begin{aligned}
x'_{k,m,n} &= x_{k,m,n} + x_{k',m',n}\frac{p_{k'',m'',n} - p_{k',m',n}}{p_{k'',m'',n} - p_{k,m,n}} \\
x'_{k',m',n} &= 0 \\
x'_{k'',m'',n} &= x_{k'',m'',n} + x_{k',m',n}\frac{p_{k',m',n} - p_{k,m,n}}{p_{k'',m'',n} - p_{k,m,n}}
\end{aligned}$$

(3) is satisfied since $x'_{k,m,n} + x'_{k',m',n} + x'_{k'',m'',n} = x_{k,m,n} + x_{k',m',n} + x_{k'',m'',n}$, and so is (2) since $x'_{k,m,n}$, $x'_{k',m',n}$, $x'_{k'',m'',n} \geq 0$ and none of them can surpass 1 or else one of the $x'\cdot,\cdot,n$ would be negative.

(2) is satisfied owing to (5), and $U' \geq U$ owing to (6). Thus we get an optimal solution where $x_{k',m',n} = 0$

In the pseudo-code below, we consider the input as points in the plane with coordinates $(p_{k,m,n}, r_{k,m,n})$. For simplicity, for a stack S, we note S[0] the top element and S[1] the second top one; for tow points A, B in the plane, we note $L(A, B)$ the slope of the line formed by them; we say B is *dominated* by A and C if and only if $A.p \leq B.p \leq C.p$ and $L(A, B) \leq L(B, C)$, which is just another interpretation of (4)

REMOVE-LP-DOMINATED(n)

```
1   find the points with p_{k,m,n} minimal, then find the one with r_{k,m,n} maximal
    amongst, noted as P_0
2   let S be a stack and PUSH(P_0, S)
3   sort the other (p_{k,m,n}, r_{k,m,n}) in an array A in decreasing order of the slope
    of the line formed between each of them and P_0
4   PUSH(A[0], S)
5   for i = 1 to A.length-1
6       if A[i].p > S[0].p
7           while L(A[i],S[0]) ≥ L(S[0],S[1])
8               POP(S)
9           PUSH(A[i], S)
10  return S
```

Proof of correctness We use the loop invariant that after each iteration of line 5, S contains exactly all the points that are not dominated by points from A[0] to A[i]

The invariant trivially holds before line 5. Suppose it holds before the ith iteration. If A[i].p $\leq$ S[0].p, by the way we sorted the points in line 3, A[i] is dominated by P_0 and S[0], and none of the points in S can be dominated by A[i] and some points from A[0] to A[i-1], so the algorithm does nothing and continue with the (i+1)th iteration.

If A[i].p $>$ S[0].p, the **while** loop removes all the points in S that are dominated by the point under it in S and A[i]. Suppose the **while** loop stops at P, that is, the top point in S after the **while** loop. This means $L(Q, P) > L(P,\text{A[i]})$ where Q is the point under P in S. In fact $L(Q, P) > L(P,\text{A[i]})$ for all Q other than P remaining in S because as Q goes down in S, $L(Q, P)$ increases by the loop invariant. Since Q is not dominated by $A \in \{\text{A[0] to A[i-1]}\}$ and P, nor is it by A and A[i]. Thus, if we push A[i] into S, the invariant still holds.

Time complexity Line 1 takes time $O(KM)$. Line 3 takes time $O(KM \log(KM))$. Regarding the **for** loop from line 5 to 9, note that each point can only be pushed or popped only once, so in all the **for** loop takes time $O(KM)$. We run the algorithm for each $n \in \mathcal{N}$, so altogether it takes time $O(NKM \log(KM))$ to remove all LP-dominated terms.