CO327 Deterministic OR Models
0. Motivation examples

- Dieting application
- Assignment problem

Andersen Ang

Dept. Combinatorics and Optimization, U.Waterloo, Canada

First draft: December 26, 2020

Last update: December 29, 2020

Dieting: application in health industry, sport industry, gym, nutrient

World Health Organization
REGIONAL OFFICE FOR Europe

Linear programming to build food-based dietary guidelines: Romanian food baskets

The American Journal of CLINICAL NUTRITION

American Society for Nutrition
ASN EST. 1928
Excellence in Nutrition Research and Practice

Linear and nonlinear programming to optimize the nutrient density of a population's diet: an example based on diets of preschool children in rural Malawi FREE

Nicole Darmon, Elaine Ferguson, André Briend

The American Journal of Clinical Nutrition, Volume 75, Issue 2, February 2002, Pages 245–253,

remember I said you can save the world?

frontiers in Nutrition

Front Nutr. 2018; 5: 48.
Published online 2018 Jun 21.
doi: 10.3389/fnut.2018.00048
PMCID: PMC6021504
PMID: 29977894

A Review of the Use of Linear Programming to Optimize Diets, Nutritiously, Economically and Environmentally

Corné van Dooren*

> Linear programming is thought to be "the ideal tool to rigorously convert precise nutrient constraints into food combinations" (3).

Linear Programming (LP) can be used to solve questions on matching diets to nutritional and other additional constraints with a minimum amount of changes. Linear programming is a mathematical technique that allows the generation of optimal solutions that satisfy several constraints at once (6).

A toy example

- You are given a list of food.
- You die if vitamin < 3000
- Q1. How to eat for lowest calories?
- Q2. How to eat for lowest spending?
- Q3. How to eat for highest tastiness?
- Q4. How to eat for lowest calories and highest tastiness?
- Q5. How to eat for lowest calories, lowest spending and highest tastiness?

Food	Vitamin	Calories	Cost	Tastiness
Apple	145	122	130	95
Banana	118	83	116	80
Mango	121	107	93	69
Orange	80	63	85	48

** These numbers are made-up*

Not to die

Food	Vitamin	Calories	Cost	Tastiness
Apple	145	122	130	95
Banana	118	83	116	80
Mango	121	107	93	69
Orange	80	63	85	48

- You die if vitamin < 3000

→ You don't die if vitamin = 3000 or vitamin > 3000

- Sol. 1. Apple only: 21 apples   21*145 = 3045 > 3000
- Sol. 2. Apple only: 9999 apples 9999*145 >>>> 3000
- Sol. 3. Orange and mango only: orange*80 + mango*121 $\geq$ 3000
- Sol. 4. Eat all of them: apple*145 + banana*118 +mango*121+ orange*80 $\geq$ 3000

Simplify the notation

	v	c	$	t
a	145	122	130	95
b	118	83	116	80
m	121	107	93	69
o	80	63	85	48

- You need $v \geq 3000$
- You eat all of them: $145a + 118b + 121m + 80o \geq 3000$
- This expression is called the **constraint set**/ **feasible set**
- (a,b,m,o) are the **optimization variables**/ **decision variable**.
- You optimize on (a,b,m,o) subject to a **goal** = minimize or maximize a **objective function**

Lowest carioles

	v	**c**	**$**	**t**
a	145	122	130	95
b	118	83	116	80
m	121	107	93	69
o	80	63	85	48

- You want c as low as possible
- You need v $\geq$ 3000
- You eat all of them: 145a + 118b + 121m + 80o $\geq$ 3000
- Total c: 122a + 83b + 107m + 63o
- The optimization problem:

$$\min_{a,b,m,o} 112a + 83b + 107m + 63o$$

$$\text{subject to } 145a + 118b + 121m + 80o \geq 3000$$

Lowest cost

	v	c	$	t
a	145	122	130	95
b	118	83	116	80
m	121	107	93	69
o	80	63	85	48

- You want $ as low as possible
- You need v ≥ 3000
- You eat all of them: 145a + 118b + 121m + 80o ≥ 3000
- Total $: 130a + 116b + 93m + 85o
- The optimization problem:

$$\min_{a,b,m,o} 130a + 116b + 93m + 85o$$

subject to $145a + 118b + 121m + 80o \geq 3000$

Highest tastiness

	v	c	$	t
a	145	122	130	95
b	118	83	116	80
m	121	107	93	69
o	80	63	85	48

- You want as tasty as possible
- You need v $\geq$ 3000
- You eat all of them: 145a + 118b + 121m + 80o $\geq$ 3000
- Total t: 95a + 80b + 69m + 48o
- The optimization problem:

$$\max_{a,b,m,o} 95a + 80b + 69m + 48o$$

$$\text{subject to } 145a + 118b + 121m + 80o \geq 3000$$

Solution to highest tastiness

- The solution to

$$\max_{a,b,m,o} 95a + 80b + 69m + 48o$$

$$\text{subject to } 145a + 118b + 121m + 80o \geq 3000$$

Answer: $a = \infty, b = \infty, m = \infty, o = \infty$

- Why: no matter what you give me (a, b, m, o), I can always find a better sol. as

$$(a + 1, b + 1, m + 1, o + 1)$$

- $a = \infty, b = \infty, m = \infty, o = \infty$ is mathematically optimal, but in reality this is nonsense.

- The nonsense comes from ***improper modelling*** of the problem.

Highest tastiness (proper modelling)

	v	c	$	t
a	145	122	130	95
b	118	83	116	80
m	121	107	93	69
o	80	63	85	48

- You want as tasty as possible
- You need v $\geq$ 3000
- You eat all of them: 145a + 118b + 121m + 80o $\geq$ 3000
- Total tastiness t: 95a + 80b + 69m + 48o
- You eat at most 5000 calories: 122a + 83b + 107m + 63o $\leq$ 5000
- You have only 10000$: 130a + 116b + 93m + 85o $\leq$ 10000
- The optimization problem:

$$\max_{a,b,m,o} 95a + 80b + 69m + 48o \text{ subject to } \begin{cases} 122a + 83b + 107m + 63o \leq 5000 \\ 130a + 116b + 93m + 85o \leq 10000 \\ 145a + 118b + 121m + 80o \geq 3000 \end{cases}$$

Even more modelling

	v	c	$	t
a	145	122	130	95
b	118	83	116	80
m	121	107	93	69
o	80	63	85	48

- You want as tasty as possible
- You need v $\geq$ 3000
- You eat all of them: 145a + 118b + 121m + 80o $\geq$ 3000
- Total tastiness t: 95a + 80b + 69m + 48o
- You eat at most 5000 calories: 122a + 83b + 107m + 63o ≤ 5000
- You have only 10000$: 130a + 116b + 93m + 85o ≤ 10000
- You cannot have negative amount of fruits: $\mathrm{a} \geq 0$, $\mathrm{b} \geq 0$, $\mathrm{m} \geq 0$, $\mathrm{o} \geq 0$
- Your mum forces you to have at least 1 apple: $\mathrm{a} \geq 1$
- Be environmentally friendly, no left-over: a, b, m, o have to be integer

$$\max_{a,b,m,o} 95a + 80b + 69m + 48o \text{ subject to } \begin{cases} 122a + 83b + 107m + 63o \leq 5000 \\ 130a + 116b + 93m + 85o \leq 10000 \\ 145a + 118b + 121m + 80o \geq 3000 \\ a \geq 0 \\ b \geq 0 \\ m \geq 0, \quad a, b, m, o \text{ have to be integer} \\ o \geq 0 \\ a \geq 1 \end{cases}$$

Simplify the notation again

$$\max_{x_1,x_2,x_3,x_4} \quad 95x_1 + 80x_2 + 69x_3 + 48x_4$$

$$s.t. \quad 122x_1 + 83x_2 + 107x_3 + 63x_4 \leq 5000$$

$$130x_1 + 116x_2 + 93x_3 + 85x_4 \leq 10000$$

$$145x_1 + 118x_2 + 121x_3 + 80x_4 \geq 3000$$

$$x_1 \geq 1 \qquad x_1 \in \mathbb{N}$$

$$x_2 \geq 0 \qquad x_2 \in \mathbb{N}$$

$$x_3 \geq 0 \qquad x_3 \in \mathbb{N}$$

$$x_4 \geq 0 \qquad x_4 \in \mathbb{N}$$

Optimization variable/ decision variable/ unknows

$$\max_{x_1, x_2, x_3, x_4} \quad 95x_1 + 80x_2 + 69x_3 + 48x_4$$

objective function / cost function

$$s.t. \quad 122x_1 + 83x_2 + 107x_3 + 63x_4 \leq 5000$$

"subject to"

$$130x_1 + 116x_2 + 93x_3 + 85x_4 \leq 10000$$

$$145x_1 + 118x_2 + 121x_3 + 80x_4 \geq 3000$$

Constraint set/

Feasible set

$$x_1 \geq 1 \qquad \boldsymbol{x_1} \in \mathbb{N}$$

$$x_2 \geq 0 \qquad \boldsymbol{x_2} \in \mathbb{N}$$

$$x_3 \geq 0 \qquad \boldsymbol{x_3} \in \mathbb{N}$$

$$x_4 \geq 0 \qquad \boldsymbol{x_4} \in \mathbb{N}$$

Linear constraints

Integer constraints

Towards systematic notation

Note

$$\begin{aligned} x_1 &\geq 1 \\ x_2 &\geq 0 \\ x_3 &\geq 0 \\ x_4 &\geq 0 \end{aligned} \quad = \quad \begin{aligned} 1x_1 + 0x_2 + 0x_3 + 0x_4 &\geq 1 \\ 0x_1 + 1x_2 + 0x_3 + 0x_4 &\geq 0 \\ 0x_1 + 0x_2 + 1x_3 + 0x_4 &\geq 0 \\ 0x_1 + 0x_2 + 0x_3 + 1x_4 &\geq 0 \end{aligned}$$

$$\max_{x_1,x_2,x_3,x_4} \quad 95x_1 + 80x_2 + 69x_3 + 48x_4$$

$$s.t. \quad 122x_1 + 83x_2 + 107x_3 + 63x_4 \leq 5000$$

$$130x_1 + 116x_2 + 93x_3 + 85x_4 \leq 10000$$

$$145x_1 + 118x_2 + 121x_3 + 80x_4 \geq 3000$$

$$1x_1 + 0x_2 + 0x_3 + 0x_4 \geq 1$$

$$0x_1 + 1x_2 + 0x_3 + 0x_4 \geq 0$$

$$0x_1 + 0x_2 + 1x_3 + 0x_4 \geq 0$$

$$0x_1 + 0x_2 + 0x_3 + 1x_4 \geq 0$$

$$x_1 \in \mathbb{N}$$

$$x_2 \in \mathbb{N}$$

$$x_3 \in \mathbb{N}$$

$$x_4 \in \mathbb{N}$$

$$=$$

$$\max_{x_1,x_2,x_3,x_4} \quad 95x_1 + 80x_2 + 69x_3 + 48x_4$$

$$s.t. \quad 122x_1 + 83x_2 + 107x_3 + 63x_4 \leq 5000$$

$$130x_1 + 116x_2 + 93x_3 + 85x_4 \leq 10000$$

$$-145x_1 - 118x_2 - 121x_3 - 80x_4 \leq -3000$$

$$-1x_1 + 0x_2 + 0x_3 + 0x_4 \leq -1$$

$$0x_1 - 1x_2 + 0x_3 + 0x_4 \leq 0$$

$$0x_1 + 0x_2 - 1x_3 + 0x_4 \leq 0$$

$$0x_1 + 0x_2 + 0x_3 - 1x_4 \leq 0$$

$$x_1 \in \mathbb{N}$$

$$x_2 \in \mathbb{N}$$

$$x_3 \in \mathbb{N}$$

$$x_4 \in \mathbb{N}$$

Canonical form

$$\max_{x_1,x_2,x_3,x_4} \quad 95x_1 + 80x_2 + 69x_3 + 48x_4$$

$$
\begin{aligned}
s.t. \quad & 122x_1 + 83x_2 + 107x_3 + 63x_4 \le 5000 \\
& 130x_1 + 116x_2 + 93x_3 + 85x_4 \le 10000 \\
& -145x_1 - 118x_2 - 121x_3 - 80x_4 \le -3000 \\
& -1x_1 + 0x_2 + 0x_3 + 0x_4 \le -1 \\
& 0x_1 - 1x_2 + 0x_3 + 0x_4 \le 0 \\
& 0x_1 + 0x_2 - 1x_3 + 0x_4 \le 0 \\
& 0x_1 + 0x_2 + 0x_3 - 1x_4 \le 0 \\
& x_1 \in \mathbb{N} \\
& x_2 \in \mathbb{N} \\
& x_3 \in \mathbb{N} \\
& x_4 \in \mathbb{N}
\end{aligned}
$$

Compact notation

$$\max_{\boldsymbol{x}\in\mathbb{N}^4} \begin{bmatrix} 95 \\ 80 \\ 69 \\ 48 \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$s.t. \begin{bmatrix} 122 & 83 & 107 & 63 \\ 130 & 116 & 93 & 85 \\ -145 & -118 & -121 & -80 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \le \begin{bmatrix} 5k \\ 10k \\ -3k \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\max_{\boldsymbol{x}\in\mathbb{N}^4} \boldsymbol{c}^T \boldsymbol{x}$$

$$s.t. \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}$$

Linear programming

$$\max_{\boldsymbol{x}\in\mathbb{R}^n} \boldsymbol{c}^T\boldsymbol{x}$$

$$s.t.\ \boldsymbol{Ax} \leq \boldsymbol{b}$$

Integer programming

$$\max_{\boldsymbol{x}\in\mathbb{N}^n} \boldsymbol{c}^T\boldsymbol{x}$$

$$s.t.\ \boldsymbol{Ax} \leq \boldsymbol{b}$$

Learning outcome of this course :

1. form this problem (modeling)
2. solve this problem (method)
3. check the solution is optimal (theory)

Motivation example 2: Assignment problem

- You are a king and you rule a kingdom.


- You have to gather 3 resources: food, wood and gold.



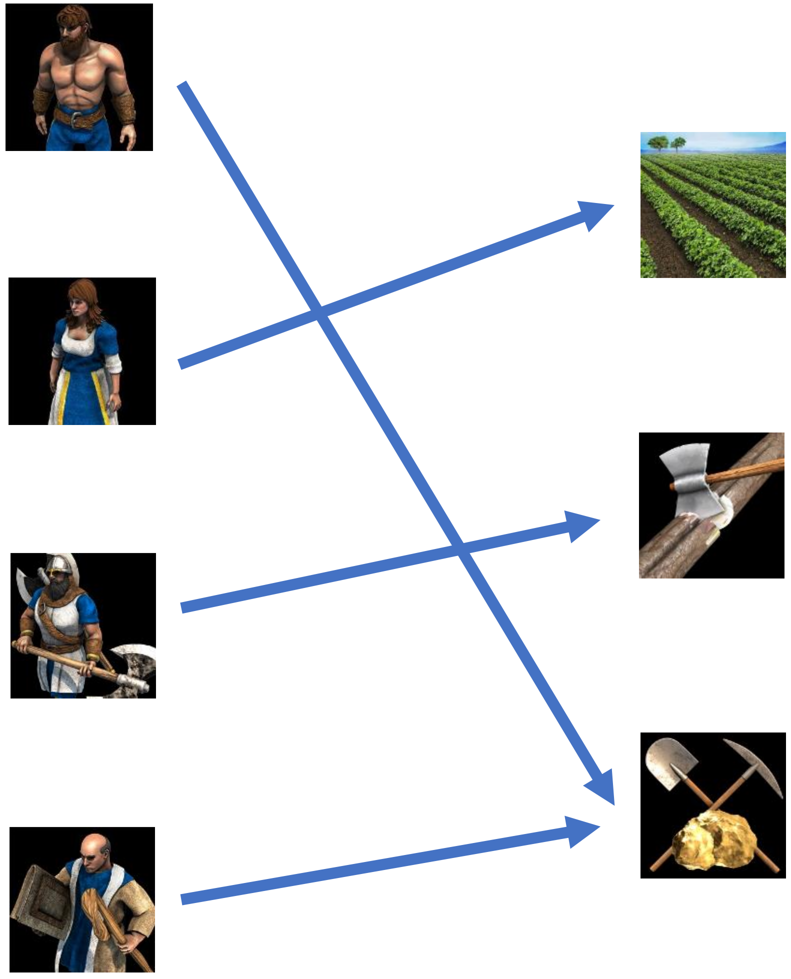
- You assign people for these jobs



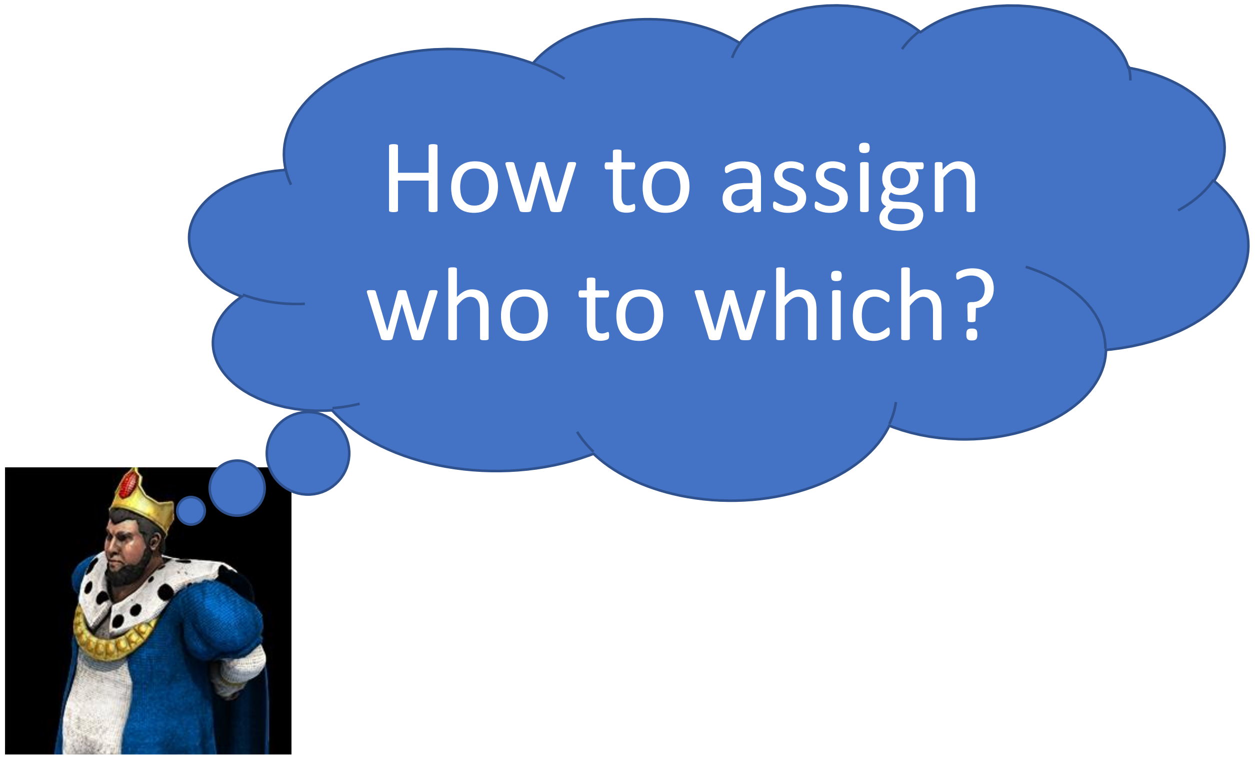

- Due to difference in experience and expertise, different person has different ability in different tasks.

In graph theory language: bipartite matching

Assignment problem: match the people with the tasks.



Applicants Job



Assignment problem: scenario 1

- Min time problem.
- A job = get 50 unit of a type of resource.
- Different person takes different amount of time to finish the job.
- You want to assign them such that all jobs are done and the total time spent is minimized.

Time required to finish the task

			
	145	122	130
	118	83	116
	121	107	93
	999	999	999

In formal language

	j_1	j_2	j_3
a_1	145	122	130
a_2	118	83	116
a_3	121	107	93
a_4	999	999	999

- A set of applicants: $A = \{a_1, a_2, a_3, a_4\}$.
- A set of jobs: $J = \{j_1, j_2, j_3\}$.
- A cost function $C: A \times J \to \mathbb{R}$
- Find a bijection $f: A \to J$ such that the total cost $\sum C(a, f(a))$ is minimized.
- Note: A and J can have equal or **unequal** sizes.

  In reality: sometime there is more people than job (e.g. university position)

Forming a solution

	j_1	j_2	j_3
a_1	145	122	130
a_2	118	83	116
a_3	121	107	93
a_4	999	999	999

- You can **fire** people: here you fire a_4.
- Now, how to solve it?
- How do you know and check your sol. is **optimal:** no other choice will be better?
- If the problem is small (like this one): you can use brute-force.

  What if the problem is 99999 x 99999 ? **How to solve it fast?**

Assignment problem: other scenarios

	j_1	j_2	j_3
a_1	145	122	130
a_2	118	83	116
a_3	121	107	93
a_4	999	999	999

- Now the table shows the amount of resources they gather in the same amount of time.
- Now assign them such that the sum is maximized.
- *You are allowed to assign two people in the same task.
- **Scenario 2a**. One job only one person.
- **Scenario 2b**. One person can do **two** jobs.
- How to solve these problems?
- Kuhn–Munkres algorithm (Hungarian algorithm)

  An algorithm that can solves the assignment problem in polynomial time.

Broad applicability of linear & integer programming

- Replace the ''calories'', ''cost'', ''tastiness'', ''time'' to other quantities, the aforementioned problems become real-life applications in other domains.
- The message here: many real-life problems are in fact the same type of problem, and they can be solved using the same type of method (LP here).
- A partial list of applications:
  - Production scheduling
  - Transportation scheduling
  - Flight crew scheduling
  - Crop scheduling
  - Dieting planning
  - Power system design
  - Supply chain management
  - Resource allocation
  - Warehousing
  - Layout design
  - Cashflow matching
  - Currency arbitrage

Google for more

Google Hash Code

- A team programming competition by Google.
- Solve (simplified) real problems from Google.
- Date: Feb each year.
- Free to participate.
- Winner get $$$, good CV (and internship opportunity to work in google?)
- You need to form at least 2 people in a team.
- Join ``local hub'' (or form a UW hub)
- Competitive coder not a must, can join for fun (like me)
- Google it for more information.

*One of your assignment *might* come from the Google Hash Code past problem.

Google Hash Code 2020

Online Qualification Round

Rank	Team Name	Country	Score
1	✷code	Japan	27203691
2	Past Glory	Russia, Belarus, Ukraine	27184696
3117	MONSter4.0	Belgium	22829311
10724	Tormentor	Ireland	0

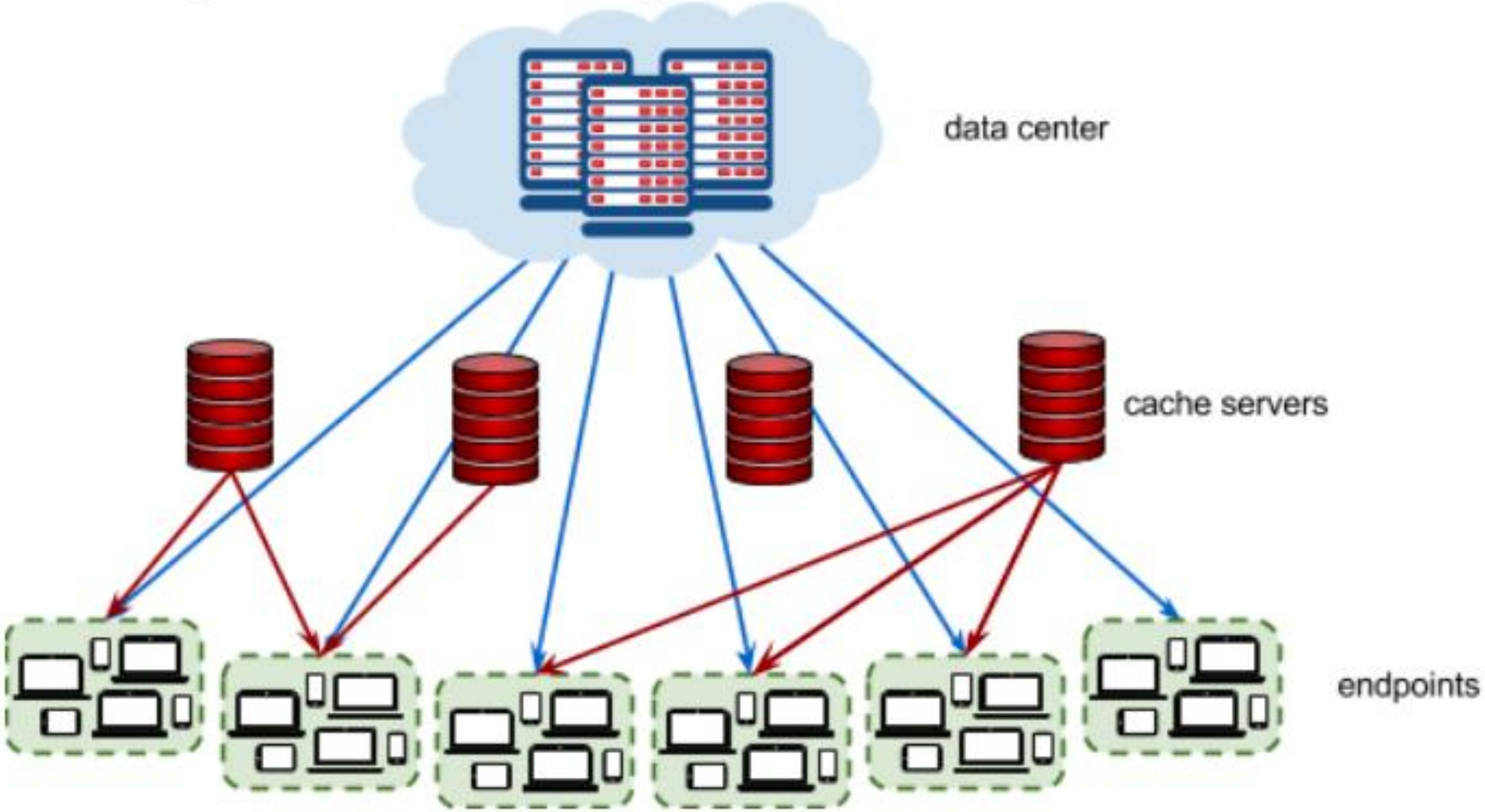
Problem from 2017

Task

Given a description of cache servers, network endpoints and videos, along with predicted requests for individual videos, **decide which videos to put in which cache server** in order to minimize the average waiting time for all requests.

Problem description

The picture below represents the video serving network.



https://codingcompetitions.withgoogle.com/hashcode/archive

Now I assume you are motivated.

Therefore no more “why study this course”.