
To make some extra cash during the semester you take up a part-time job at CUHK. On your first morning of work, your boss – let's call him Bob – gives you an inventory of the watermelon reserves across campus:

location	stock
330 Cafe	3 watermelons
S. H. Ho Canteen	1 watermelon
Morningside Canteen	7 watermelons
University Guesthouse	2 watermelons
Canteen at Medical School	1 watermelon

In anticipation of the lunch hour rush, Bob would like to redistribute the watermelons like this:

location	request
330 Cafe	2 watermelons
S. H. Ho Canteen	4 watermelons
Morningside Canteen	1 watermelon
University Guesthouse	1 watermelon
Canteen at Medical School	6 watermelons

The distance between any pair of consecutive locations, give or take, is about 100 meters.

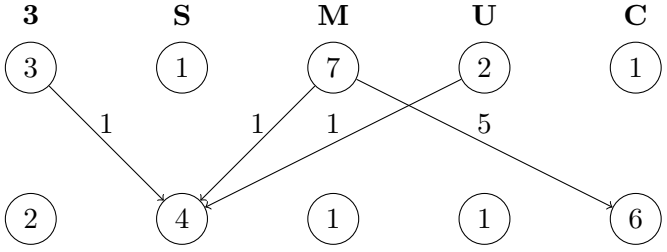
$$\mathbf{3} \xleftrightarrow{100\text{m}} \mathbf{S} \xleftrightarrow{100\text{m}} \mathbf{M} \xleftrightarrow{100\text{m}} \mathbf{U} \xleftrightarrow{100\text{m}} \mathbf{C}$$

Shuttling watermelons is demanding work. A porter asks for 10 HKD for every watermelon carried over a distance of 100 meters. If he were to carry, for example, two watermelons from the 330 Cafe to Morningside – at a distance of 200 meters – that would cost Bob 40 HKD. Being on a tight budget, Bob asks his other employee Jason, a student in the Business School, to come up with an economical way of moving the watermelons around.

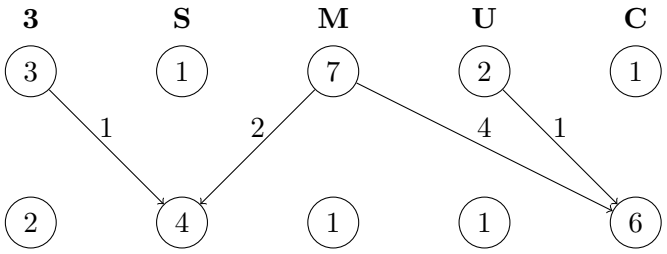
You, a promising young engineer in pursuit of an exciting assignment, overhear Jason shouting the following instructions to the porter:

> Move 5 watermelons from Morningside to the Canteen at the Med School. Then take one watermelon from each of the 330 Cafe, Morningside, and the University Guesthouse and move those to S. H. Ho.

Jason's plan involves moving two watermelons at a distance of 100 meters and another six at a distance of 200 meters, for a total cost $2 \times 10 + 6 \times 20 = 140$ dollars. You take out a piece of paper and make a quick sketch of it:



You notice something fishy: The trips out of Morningside and the University Guesthouse cross paths. This is clearly wasteful, so why not move the watermelons around like this instead:



Indeed, now you are moving 4 watermelons at 100 meters and another 4 at 200 meters for a total cost of 120 calories! Bursting with excitement, you go and show your improvement to Bob. He is happy that you will save him 20 HKD. Then he asks: "Since you are so bright, can you save me another 20 HKD and do the whole operation for 100?"

You scratch your head for a few minutes, but you are at a loss; nothing seems to work. You are a bit embarrassed to admit your failure to Bob. What if your rival Jason impresses Bob with an even better solution?

Proofs

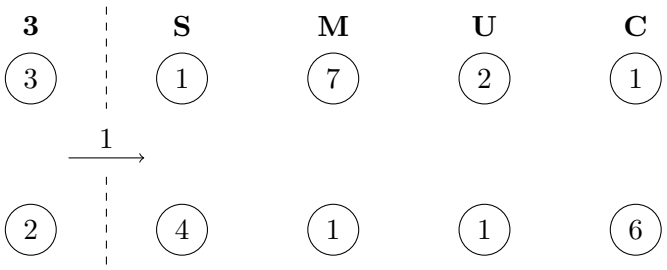
Much of the mathematics you study in school is about *calculating* things. In first grade you learn how to add single digit numbers. Later you move on to larger numbers and more complicated calculations, like multiplications, divisions, and square roots. In high school, you may be calculating roots of quadratic equations, sines and cosines, and doing some complex number algebra. At university, you take derivatives and compute integrals, solve systems of linear equations and differential equations.

In order to be a good engineer, you certainly need to be a master at various calculations that come up routinely in your discipline. But calculating is not enough. You will need to learn to set up and solve problems in a confident manner.

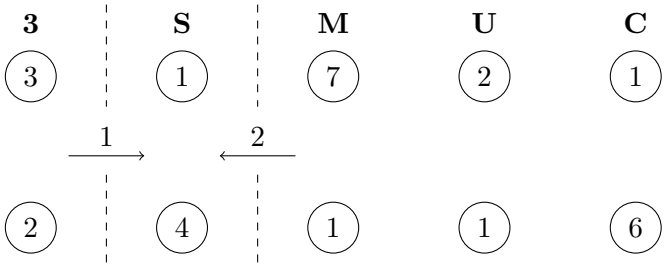
In Bob's watermelon transportation task, a mediocre engineer (Jason) might be satisfied with a solution that "feels" good to him. A great engineer, like you, wants more: You want to be sure that your solution is the best possible one. For this, calculations are not particularly helpful; you need to reason things out.

After thinking for a while, you are quite sure that it is impossible to spend less than 120 dollars on the watermelons. But how do you explain this to Bob?

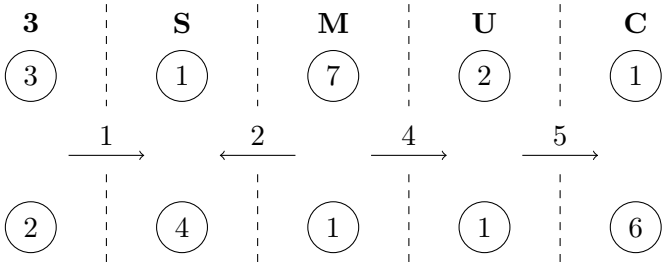
Here is how. You notice that at the 330 Cafe, there are three watermelons in the morning and we need to end up with two in the afternoon; so *no matter how things are moved*, at least one watermelon will have to be carried out of the 330 Cafe:



Next, if you add the number of watermelons at the 330 Cafe and S. H. Ho, there are 4 available but 6 are needed; so no matter how the watermelons are carried, at least two will have to be brought in from Morningside or beyond:



Continuing your reasoning in this way, you come up with the following picture:



It is now clear that Bob must spend at least 10 HKD moving crates between the 330 Cafe and S. H. Ho, at least 20 HKD between S. H. Ho and Morningside, and so on. Adding all the expenses together amounts to exactly 120 HKD. Your solution was indeed the best possible.

What we just saw is an example of a *proof*: A deduction of an interesting *proposition* ("No matter how the watermelons are carried, Bob must pay at least 120 HKD") by a sequence of clear, rigorous *logical deductions*. The ability to come up with proofs and present them clearly is important for computer science and other engineering disciplines. In the next few weeks, we will talk about various types of proofs in some detail.

1 Propositions

A *proposition* is a statement that is either true or false. Here are two examples of propositions.

> $1 + 1 = 2$.
> Thursday is the day after Tuesday.

The first proposition is about numbers; the second one is about days of the week. The first proposition is true; the second one is false. This can be figured out by most people with a first grade education (where we all learned the meaning of "1", "+", "Tuesday", and so on).

Telling whether a proposition is true or false is not always easy, but the *meaning* of a proposition must always be clear. For example, consider the following statements:

> Smoking kills.
> I will always love you.
> You may have cake, or you may have ice cream.

What do they mean exactly? Who does smoking kill? Will you love me even if I moved to Australia for good? May I have ice cream on top of my cake? In daily life, this kind of ambiguity is tolerable and even desirable, but it is not acceptable in mathematics and much of computer science (in programming, for example). We will not call such statements propositions.

Propositional logic and truth tables

We can modify and combine propositions using *operators* such as AND, OR, and NOT. For example, the proposition

$$\text{NOT } (1+1=3)$$

is true, while the proposition

$$(1+1=2) \text{ AND } (1+1=3)$$

is false.

In general, given an arbitrary proposition P, we can build the proposition NOT P. The proposition NOT P is false when P is true, and true when P is false. We can describe the effect of NOT compactly in a *truth table*:

P	NOT P
T	**F**
F	**T**

Given two propositions P and Q, we can form the *compound propositions* P AND Q, P OR Q. Here are their truth tables:

P	Q	P AND Q
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**F**

P	Q	P OR Q
T	**T**	**T**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

This is a different from the way the word "or" is used in common English. When you see a dinner set in a restaurant that comes with "beer or wine", it is usually understood that you cannot have both. In contrast, in mathematics and computer science, P OR Q is also true when P is true and Q is true.

The English meaning of "You can have beer or wine with your dinner" is captured by the operator XOR, which stands for "exclusive or":

P	Q	P XOR Q
T	**T**	**F**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

We say P and Q are *logically equivalent* if they take the same truth value. The operator IFF (short for "if and only if") describes logical equivalence:

P	Q	P IFF Q
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**T**

We can go on and on, but in fact every compound proposition is logically equivalent to a propositional formula that uses only the operators AND, OR, and NOT. For example

$$P \text{ XOR } Q \quad \text{is logically equivalent to} \quad ((\text{NOT } P) \text{ AND } Q) \text{ OR } (P \text{ AND } (\text{NOT } Q)).$$

One way to verify this is to compute the truth table of the second formula and compare it to the truth table for XOR:

P	Q	NOT P	(NOT P) AND Q	NOT Q	P AND (NOT Q)	((NOT P) AND Q) OR (P AND (NOT Q))
T	**T**	**F**	**F**	**F**	**F**	**F**
T	**F**	**F**	**F**	**T**	**T**	**T**
F	**T**	**T**	**T**	**F**	**F**	**T**
F	**F**	**T**	**F**	**T**	**F**	**F**

2 Quantifiers

A *predicate* is a proposition whose truth may depend on one or more *free variables*. For example, "n is even" is a predicate (about integer numbers) with free variable n. It is true when $n = 2$ and false when $n = 3$. The predicate "$n = 2 \times m$" is true when $n = 4, m = 2$ and false when $n = 2, m = 4$.

A predicate can be turned into a proposition by *quantifying* over the free variables. For example, the statement

> For all n, n is even

is a proposition. This proposition is false because when $n = 3$, "n is even" becomes false. On the other hand, the proposition

> There exists an n such that n is even

is true because when $n = 2$, "n is even" becomes true.

A proposition like "10 is even" can itself be written using quantifiers: A number is even if it equals twice *some* other number, namely

> There exists an m such that $10 = 2 \times m$.

This proposition is true because $10 = 2 \times 5$. The proposition "For all numbers n, n is even" can be written as

For all n there exists an m such that $n = 2 \times m$

As we saw, this one is false.

Both the *names* of the quantified variables and the *order* in which they appear matters in such statements. Do not be careless with them! For example, if we change the role of m and n, we obtain the proposition

For all m there exists an n such that $n = 2 \times m$

which is true. Now if we change the order of the quantifiers in the last statement, we obtain

There exists an n such that for all m, $n = 2 \times m$

which is false again.

The implies operator

The IMPLIES operator, which we also write as $\longrightarrow$, captures the meaning of the English conditional "If P then Q". It has the following truth table:

P	Q	$P \longrightarrow Q$
T	**T**	**T**
T	**F**	**F**
F	**T**	**T**
F	**F**	**T**

So the proposition

$$1 + 1 = 2 \longrightarrow 1 + 1 = 3$$

is false. On the other hand, the propositions

$$1 + 1 = 3 \longrightarrow 1 + 1 = 2$$
$$1 + 1 = 3 \longrightarrow 1 + 1 = 4$$

are both true. This may sound a bit strange at first, but it makes perfect sense: If $1 + 1 = 3$, which is clearly false, then anything goes. *A false proposition implies any other proposition.*

The *implies* operator comes in handy when reasoning about predicates. For example, let's take the following proposition about integer numbers:

(1) Every even number is the sum of two odd numbers.

Let's give the number a name; let's call it n. Proposition (1) says that if n happens to be even,