Math 408

L-functions and Sphere Packing

Steven Miller: sjm1@williams.edu

https://web.williams.edu/Mathematics/sjmiller/public_html/408Fa20/

Lecture 34: December 7, 2020

The Kepler Conjecture

The Kepler Conjecure:

Kepler conjecture

From Wikipedia, the free encyclopedia

The **Kepler conjecture**, named after the 17th-century mathematician and astronomer Johannes Kepler, is a mathematical theorem about sphere packing in three-dimensional Euclidean space. It states that no arrangement of equally sized spheres filling space has a greater average density than that of the cubic close packing (face-centered cubic) and hexagonal close packing arrangements. The density of these arrangements is around 74.05%.

In 1998 Thomas Hales, following an approach suggested by Fejes Tóth (1953), announced that he had a proof of the Kepler conjecture. Hales' proof is a proof by exhaustion involving the checking of many individual cases using complex computer calculations. Referees said that they were "99% certain" of the correctness of Hales' proof, and the Kepler conjecture was accepted as a theorem. In 2014, the Flyspeck project team, headed by Hales, announced the completion of a formal proof of the Kepler conjecture using a combination of the Isabelle and HOL Light proof assistants. In 2017, the formal proof was accepted by the journal *Forum of Mathematics, Pi*.[1]

Born	27 December 1571 Free Imperial City of Weil der Stadt, Holy Roman Empire
Died	15 November 1630 (aged 58) Free Imperial City of Regensburg, Holy Roman Empire
Nationality	German
Education	Tübinger Stift, University of Tübingen (M.A., 1591)[1]
Known for	Kepler's laws of planetary motion Kepler conjecture *Rudolphine Tables*

History

- Kepler: 1611: On the six-cornered snowflake:
  - See https://www.nature.com/articles/480455a for commentary.
  - See http://www.joostwitte.nl/M_Galilei/Johannes_kepler_snowflake.pdf for text.
  - Proof announced 1998: https://annals.math.princeton.edu/wp-content/uploads/annals-v162-n3-p01.pdf

- Fermat’s Last Theorem: stated in 1637, solution published in 1995.
  - Summary: https://www.ams.org/notices/199507/faltings.pdf
  - *Richard Taylor and Andrew Wiles (May 1995). "Ring-theoretic properties of certain Hecke algebras". Annals of Mathematics. **141** (3): 553–572. https://en.wikipedia.org/wiki/JSTOR_(identifier)*
  - *Wiles, Andrew (1995). "Modular elliptic curves and Fermat's Last Theorem". Annals of Mathematics. **141** (3): 443–551. https://en.wikipedia.org/wiki/JSTOR_(identifier)*
  - *Good links: http://math.albany.edu/g/Math/topics/fermat/*

Geometry

Images from my book on Operations Research

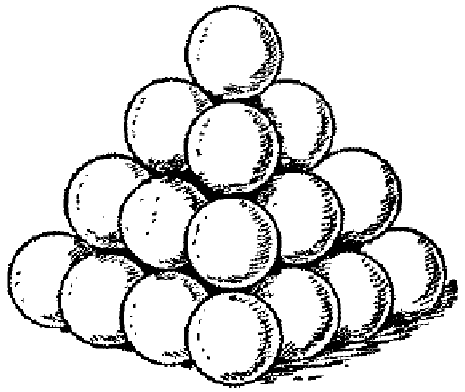


Figure 1. Thomas Harriot's Cannonball Packing Scheme. (From *Good Old-Fashioned Challenging Puzzles* (page 28), H. E. Dudeney, © Summersdale Publishers Ltd. 2007.)

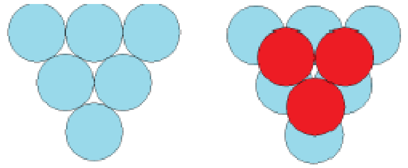


Figure 3. Aerial view of the Face-centered Cubic Packing (FCC). The spheres are placed in hexagonal patterns in rows, with the centers of one row directly above the holes of the previous.

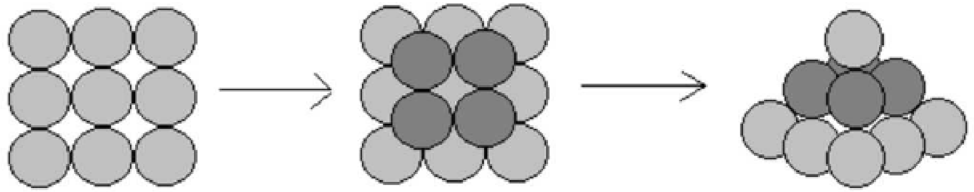


Figure 4. Hexagonal Close Packing (HCP) (image from [H2]).

A first difficulty is defining a rigorous notion of density for sphere packing. Up until now we have talked about density in mostly vague terms. The conjecture concerns the density of a packing arrangement that fills all of three-dimensional Euclidian space. At the same time, we have not yet mentioned how one can go about defining density of a packing. This problem of defining density for sphere packings is resolved using limiting notions of packing a large region and letting its diameter increase to infinity. As natural as it seems, this notion was not clarified until the 20th century. One can then prove a result asserting that a packing exists that attains a maximal limiting density.

A second difficulty is that these notions of limiting density are very crude in the sense that one can always remove spheres from an arbitrarily large finite region without changing the limiting density. Therefore we wish to impose further local restrictions on our notion of a "densest" packing. We require from our packing that it contain no large "holes." Hales and Ferguson formulated this restriction by defining a saturated packing of spheres to be one into which no new sphere can be inserted, i.e., there is no "hole" of diameter 2 or larger in the packing. Another related problem is that one might be able to increase density locally by removing a finite collection of spheres in a region and repacking that region to squeeze in one more sphere. This sort of condition is more difficult to analyze, but it already motivates the study of "local" conditions specifying density of a packing [BBC].

A third difficulty is that there exist uncountably many "optimally dense" packings that are, strictly speaking, different from each other. Here we consider packings as essentially different if they are not congruent under a Euclidean motion of space [**Lag3**]. Consider the packing that starts with a planar layer of hexagonally close packed spheres. That is, there is a planar slice through this layer that intersects all the sphere centers, giving a circle packing of the plane, and this packing is the optimal two-dimensional hexagonal circle packing. Now one can fit a second identical layer of spheres on top of this layer, so that the spheres fit as deep as possible in indentations between the spheres in the first layer. If we mark one sphere as the center of the first layer, there are two possible ways to do this: each such packing

A **Voronoi cell** around a given sphere center is the set of all points in space closer to that sphere center than to any other sphere center. In the case of the packing arrangements using hexagonal packing layers that we just described, all Voronoi cells have one of two shapes, each having 12 faces and 14 vertices. Each of these shapes is a type of dodecahedron (see [**Lag3**] for an illustration).

Another difficulty is posed by the fact that the optimization problem is essentially infinite dimensional. There are an infinite number of spheres to pack and therefore there are an infinite number of coordinates that need to be determined. The approaches that are aimed at dealing with this issue of infinite-dimensionality seek to prove stronger results which only involve finite-dimensional optimizations that encode local conditions. This is called the **local density inequality approach**. The underlying idea is to assign, by some recipe, to each sphere in a sphere packing a (weighted) sum of the covered and uncovered volume near that sphere center. This recipe is said to be "local" because the weighted sum for a given sphere center is completely determined by the locations of all spheres in the sphere packing nearby, within a fixed distance C of the given sphere center. When the recipe quantities are added up over all spheres in a given sphere packing, it should count all volume with weight 1. As a result, an upper bound on the weighted area will give an upper bound on global sphere packing density. We call such a local density inequality "optimal" if it produces the Kepler upper bound for density of sphere packing, i.e., $\pi/\sqrt{18}$.

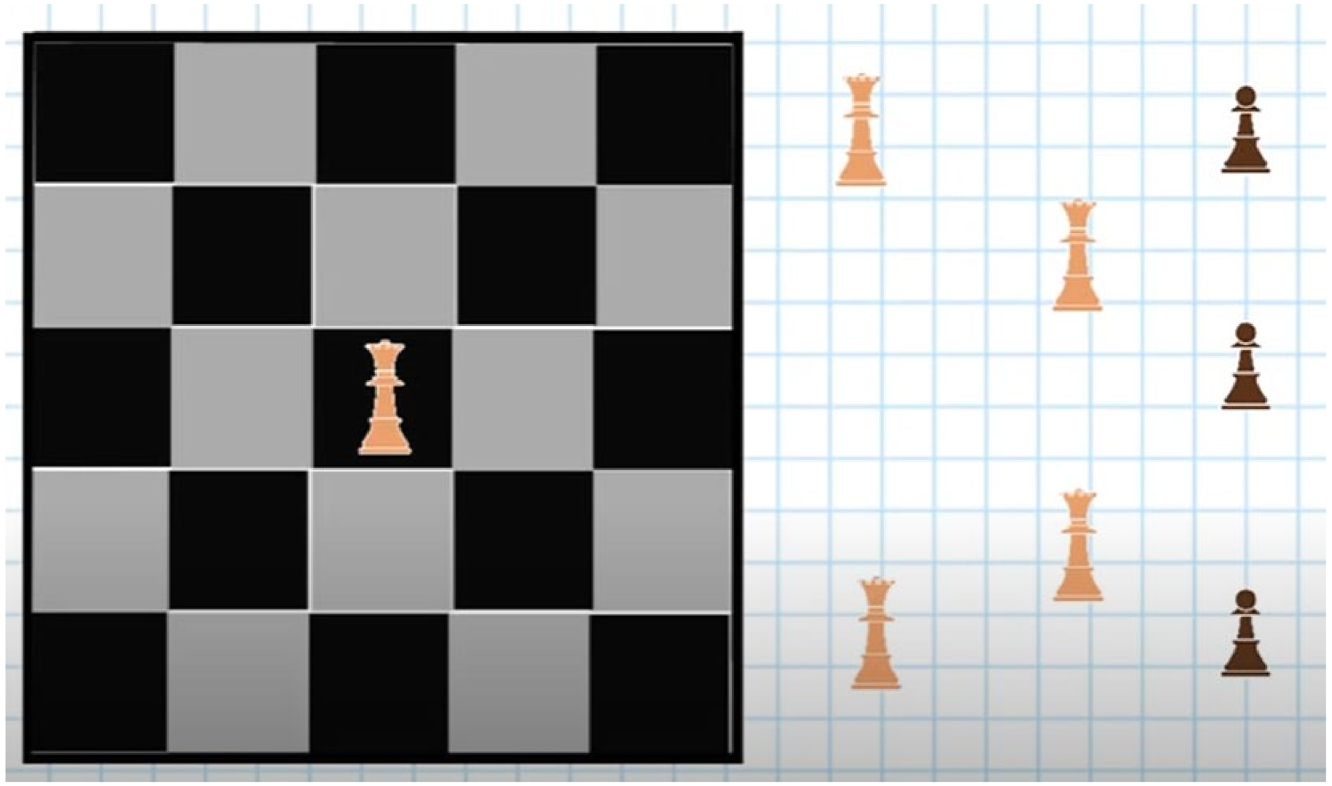
Proof of inequalities via interval arithmetic: A key element of the Hales-Ferguson proof was the use of computers to do **interval arithmetic** [**H3**]. The goal was to reduce very large computations to relatively small calculations and to ensure that there are no errors. Computers were used to prove various inequalities in a small number of variables using interval arithmetic. The computers produced upper and lower bounds for a desired value, the score function in our case, in significantly less time than it would have taken to compute an exact result. This significantly reduced the computational time needed to prove all the inequalities needed for the proof. Explicitly, the goal

Linear Programming Bounds: This was perhaps the part of the proof where the help of computers was of greatest importance, and the main reason we have included a discussion of this topic. Hales and Ferguson reduced the problem to a finite number of finite-dimensional problems. In practice, however, these problems were still out of the reach of humans because they involved about 150 variables and because of the very large number of individual problems. Thankfully, modern computers were able to save the day yet again, just like they did with the four color theorem. Many of the nonlinear optimization problems for the scores of decomposition stars are replaced by linear problems that dominate the original score. They then used computers to solve linear programming problems. A typical one has between 100 and 200 variables and 1000 and 2000 constraints. Nearly 100,000 such problems enter into the proof [**Lag3**]. When linear programming methods do not give sufficiently good bounds, they have been combined with branch and bound methods from global optimization. In addition to all of the computations, optimizations, and combinatorics, computers were also employed by Hales and Ferguson to organize their output. The organization of the few gigabytes of code and data that enter into the proof was in itself a nontrivial undertaking.

Chess Problem:
https://www.youtube.com/watch?v=aMorr1h4Egs

Place 5 queens on a 5x5 board so that 3 pawns can be safely placed.

How should we explore all the possibilities efficiently?

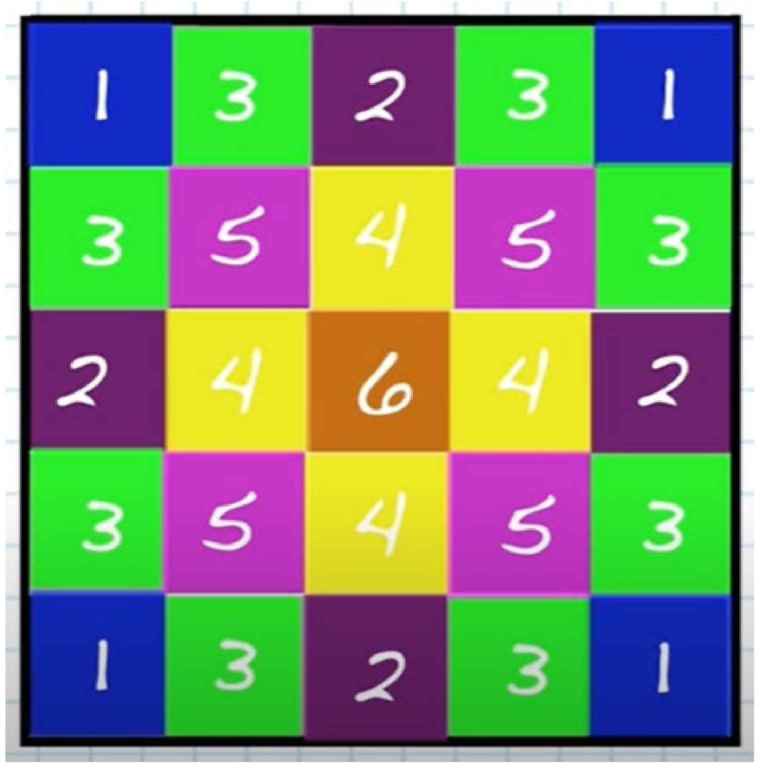


Chess Problem:
https://www.youtube.com/watch?v=aMorr1h4Egs

Place 5 queens on a 5x5 board so that 3 pawns can be safely placed.

How should we explore all the possibilities efficiently?

Can we do better?



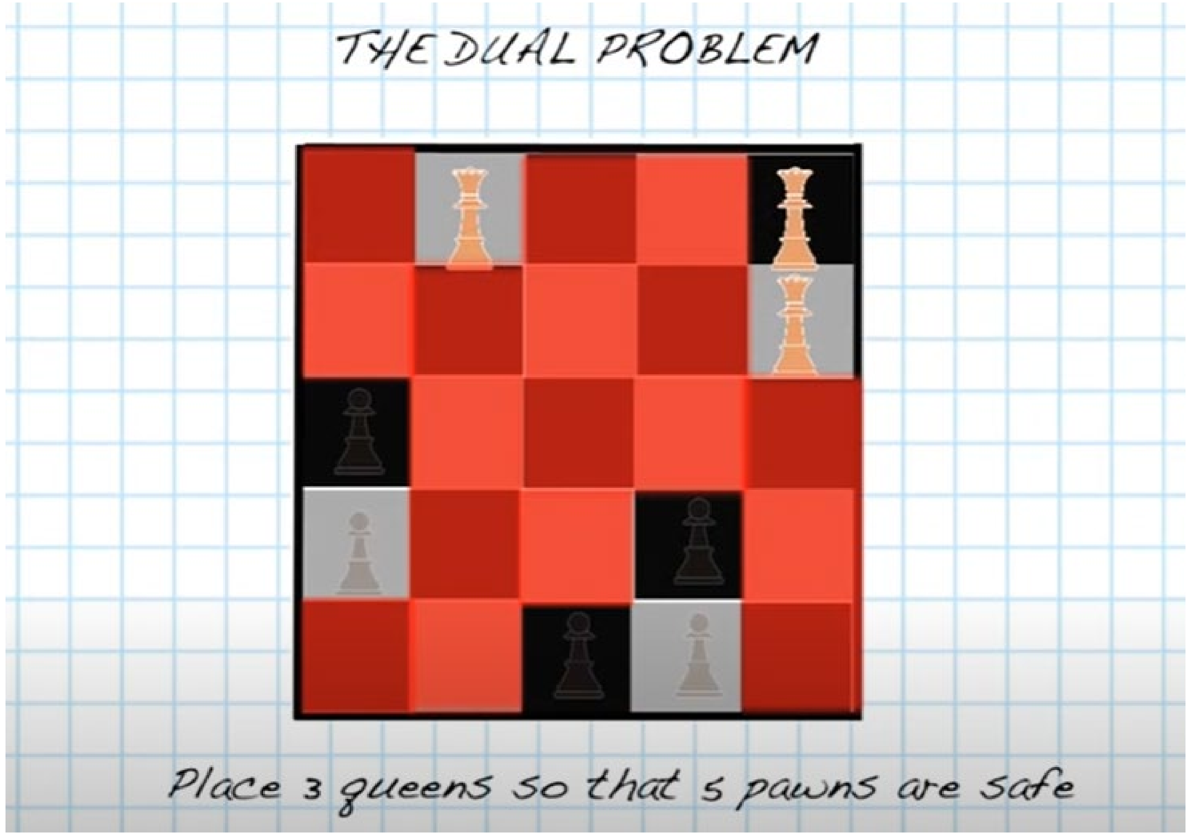
1	3	2	3	1
3	5	4	5	3
2	4	6	4	2
3	5	4	5	3
1	3	2	3	1

Chess Problem:
https://www.youtube.com/watch?v=aMorr1h4Egs

Place 5 queens on a 5x5 board so that 3 pawns can be safely placed.

How should we explore all the possibilities efficiently?

What does this remind you of?



Definition 4.3.1 (Canonical Linear Programming Problem). *The canonical Linear Programming problem has the following form:*

(1) We have variables $x_j \geq 0$ for $j \in \{1, \ldots, N\}$.

(2) The variables satisfy linear constraints, which we can write as $\mathbf{A}\vec{x} = \vec{b}$ (where $\mathbf{A}$ is a matrix with M rows and N columns, and $\vec{b}$ is a column vector with M components).

(3) The goal is to minimize a linear function of the variables: $\vec{c}^{\,T}\vec{x} = c_1x_1 + \cdots + c_Nx_N$.

- Seen canonical forms before: solving polynomials!
- Generalizing linear algebra: $\mathbf{A}\vec{x} = \vec{b}$. Difference is that we have constraints on the variables and we have a linear function we wish to minimize. Over or under-determined?

Dual Problem: Given a canonical linear programming problem, its Dual Problem is the following.

- Constraints: $\vec{y}^T \mathbf{A} \leq \vec{c}^{\,T}$, $y_j \in \mathbb{R}$.
- Objective function: maximize $\vec{y}^T \vec{b}$.

Lemma 5.2.1. *Let $\hat{x}$ be a feasible solution to the canonical linear programming problem, and let $\hat{y}$ be a feasible solution to the dual problem. Then*

$$\hat{y}^T \vec{b} \leq \vec{c}^{\,T} \hat{x},$$

and if equality holds then $\hat{x}$ is optimal.

Diet Problem: two foods. Imagine we have two foods and two nutrients, where one unit of the first has a_{11} units of the first nutrient and a_{21} of the second, while our second food has a_{12} units of the first nutrient and a_{22} of the second. Assume we need b_i units of the nutrient i each day to survive, and the cost of food k is c_k dollars per unit. Our goal is to minimize

$$c_1 x_1 + c_2 x_2$$

subject to the condition that

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad x_1, x_2 \geq 0.$$

We can rewrite this as

$$\text{Goal}: \ \text{minimize } \vec{c}^{\,T} \vec{x} \text{ subject to } \mathbf{A}\vec{x} \geq \vec{b} \text{ and } \vec{x} \geq \vec{0}.$$

We're trying to minimize the cost $20x_1 + 2x_2$. Consider the linear function $\mathrm{Cost}(x_1, x_2) = 20x_1 + 2x_2$. We look at contours where the function is constant: $\mathrm{Cost}(x_1, x_2) = c$; we sketch some constant cost contours in Figure 5.

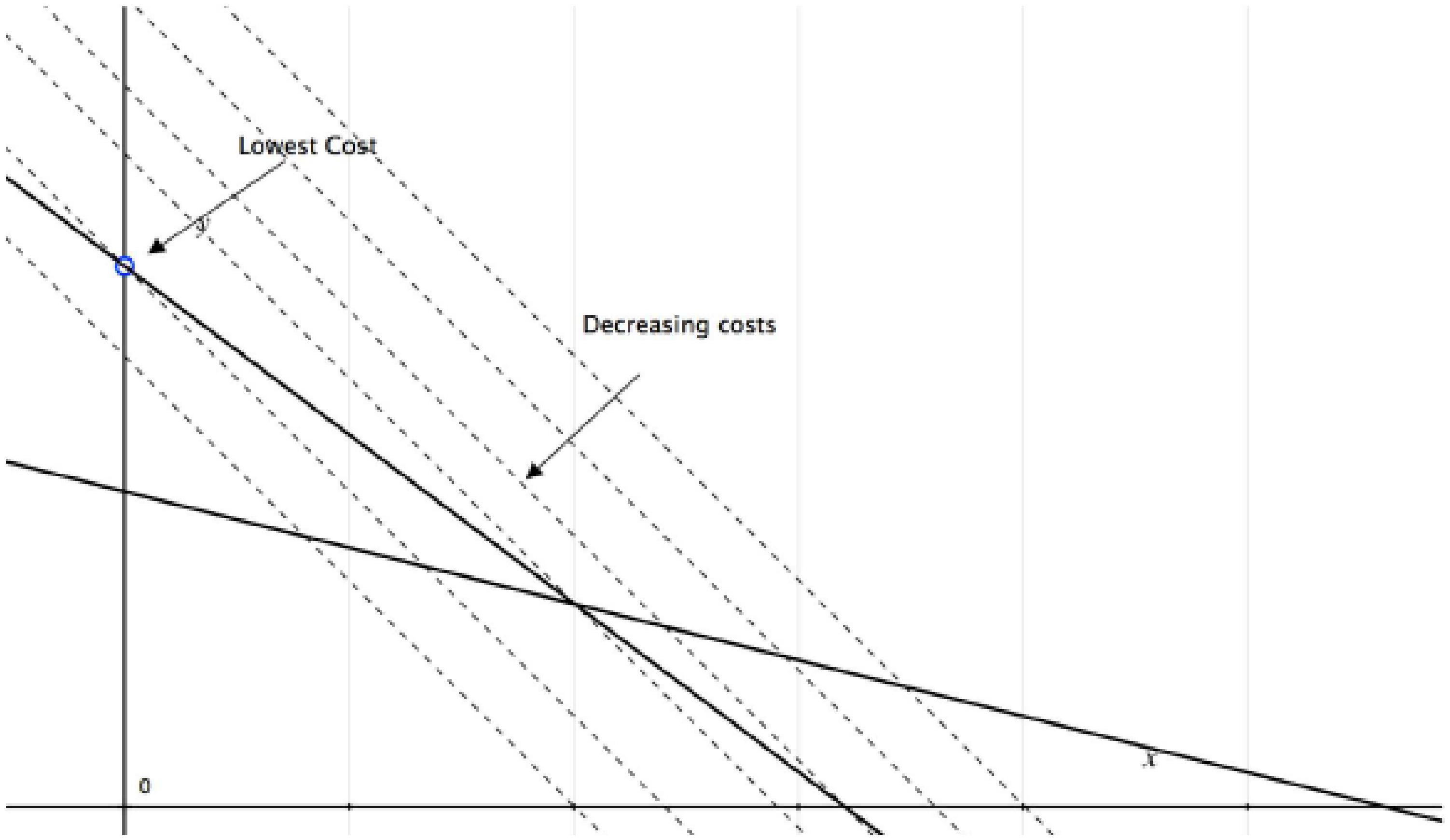


Figure 5. Solving the diet problem. Problem.

Power of Duality:

For functions f with suitable decay, we have the **Poisson Summation Formula:**

$$\sum_{n=-\infty}^{\infty} f(n) \; = \; \sum_{n=-\infty}^{\infty} \widehat{f}(n);$$

f being in the Schwartz space suffices. Poisson summation is often used to convert a slowly decaying sum, where we need to take many terms to obtain a good approximation, to another sum with rapid decay, where just a few terms (sometimes even just one term) provides an excellent approximation. The next problem gives an important example; that result plays a key role in a variety of subjects, such as the Riemann zeta function and the distribution of primes and Benford's law of digit bias.

Exercise 5.4.30. *Prove*

$$\frac{1}{\sqrt{N}} \sum_{n=-\infty}^{\infty} e^{-\pi n^2/N} \; = \; \sum_{n=-\infty}^{\infty} e^{-\pi n^2 N}.$$

So far all the variables in our linear programming problems are real, taking values from a continuum. For many problems this is reasonable, though at first glance it might not appear so. For example, we might worry about being able to ship any amount of oil (perhaps we can only send an integral number of gallons), or charge any amount (for most products the cost must be some number of cents); however, if the quantities are large then for all practical purposes there's no harm in allowing these quantities to vary continuously. If we're looking at the cost of a flight, which is measured in hundreds of dollars, a hundredth of a cent is immaterial; if we're shipping oil on a tanker, which can transport millions of gallons, part of a gallon will not be noticeable.

Unfortunately, in other problems we cannot make such a simplifying assumption. For example, if we consider the Diet Problem and we only have a few items, you can't purchase just part of a product and leave the rest (as Kramer found out when we wanted to buy part of a can of coke and part of an apple in the Seinfeld episode *The Seven*, or as Steve Martin's character found out in the movie *Father of the Bride* with hot dog buns). In these settings, we need to restrict to integer amounts; not surprisingly, this leads to the subject of **Integer Programming**.

https://www.youtube.com/watch?v=r-C82iNY8jk https://www.youtube.com/watch?v=oYIHLUxzRr8

Assumptions for Integer Programming: For problems involving integer variables, we assume the following always hold.

(1) Any quantity A under consideration is **bounded**; this means there is some number N such that $|A| \leq N$. We frequently include a subscript and write N_A for the bound to highlight its dependence on A.

(2) We assume our quantities are discrete with a fixed smallest unit (which everything else is an integral multiple). We often denote this small quantity by δ.

Theorem 8.2.1. *Consider a quantity A such that $-N_A \leq A \leq N_A$ (i.e., $|A|$ is at most N_A), and A is discrete (i.e., $A \in \{0, \pm\delta, \pm 2\delta, \dots\}$). The following constraints ensure that z_A is 1 if $A \geq 0$ and z_A is 0 otherwise:*

(1) $z_A \in \{0, 1\}$.

(2) $\frac{A}{N_A} + \frac{\delta}{2N_A} \leq z_A$.

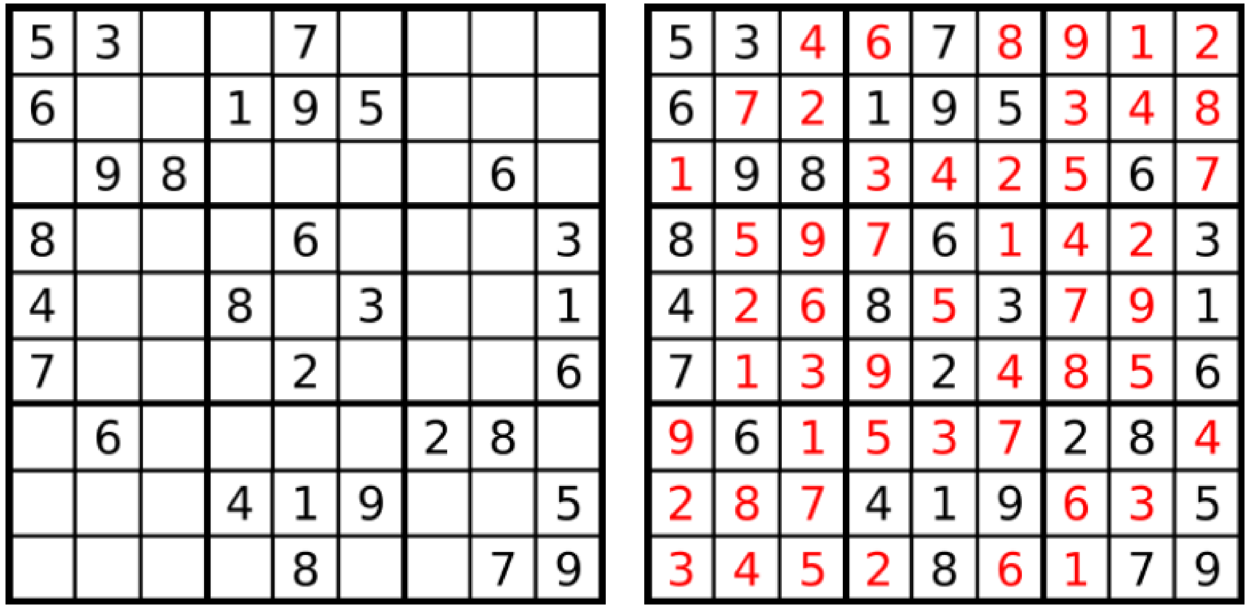
(3) $z_A \leq 1 + \frac{A}{N_A}$.

What can Linear Programming handle?

- Are we above / below a threshold.
- Logical operations: and, if-then, or, xor.
- Trunctation.
- Max/min and absolute values.

Efficient Programming

How many constraints to check?



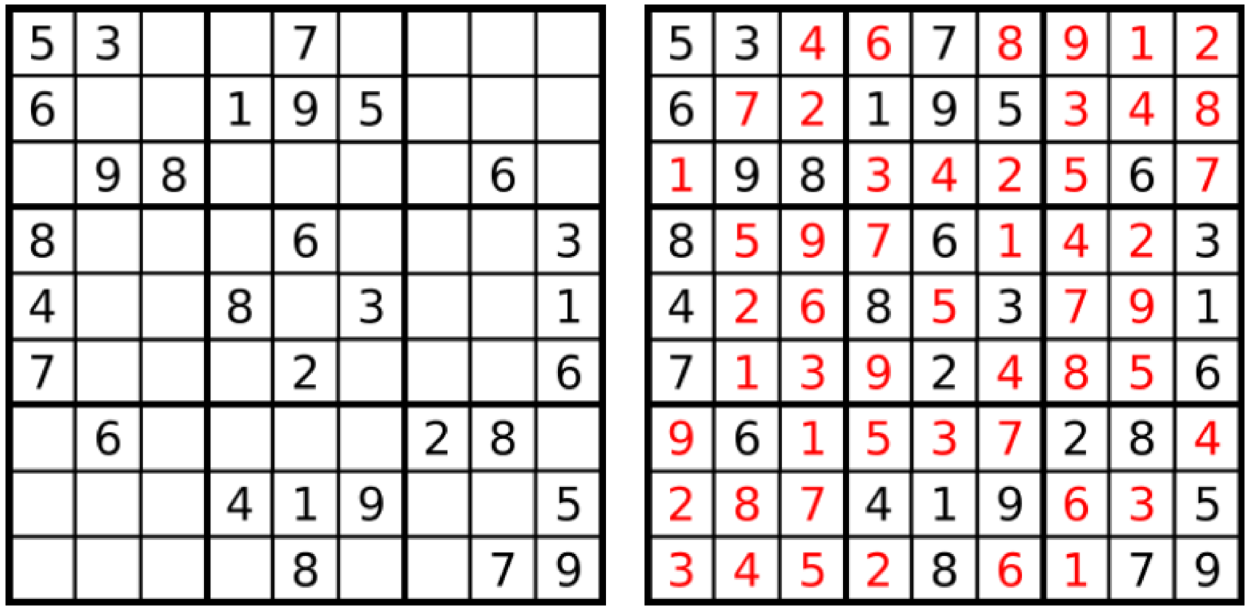
5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Figure 1. An example of a Sudoku problem (left) and its solution (right). Images from Wikimedia Commons (left from Lawrence Leonard Gilbert, right from Colin M. L. Burnett; licensed under the Creative Commons Attribution Share Alike 3.0 Unported (`https://creativecommons.org/licenses/by-sa/3.0/deed.en`) license).

Efficient Programming

How many constraints to check?



5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Figure 1. An example of a Sudoku problem (left) and its solution (right). Images from Wikimedia Commons (left from Lawrence Leonard Gilbert, right from Colin M. L. Burnett; licensed under the Creative Commons Attribution Share Alike 3.0 Unported (https://creativecommons.org/licenses/by-sa/3.0/deed.en) license).

A clever choice makes formulating the constraints significantly easier:

$$w_{ij} \in \{1, 10, 100, 1000, \dots, 10^8\}.$$

Why is this better? Notice the only way to have nine numbers from this list sum to 111,111,111 is to have one of each. Thus, instead of a massive use of `OR` statements we could do

$$\forall i \in \{1, \dots, 9\} : \sum_{j=1}^{9} x_{ij} = 111,111,111.$$

Linear Programming: Harder than Non-Linear

Traveling Salesman Problem

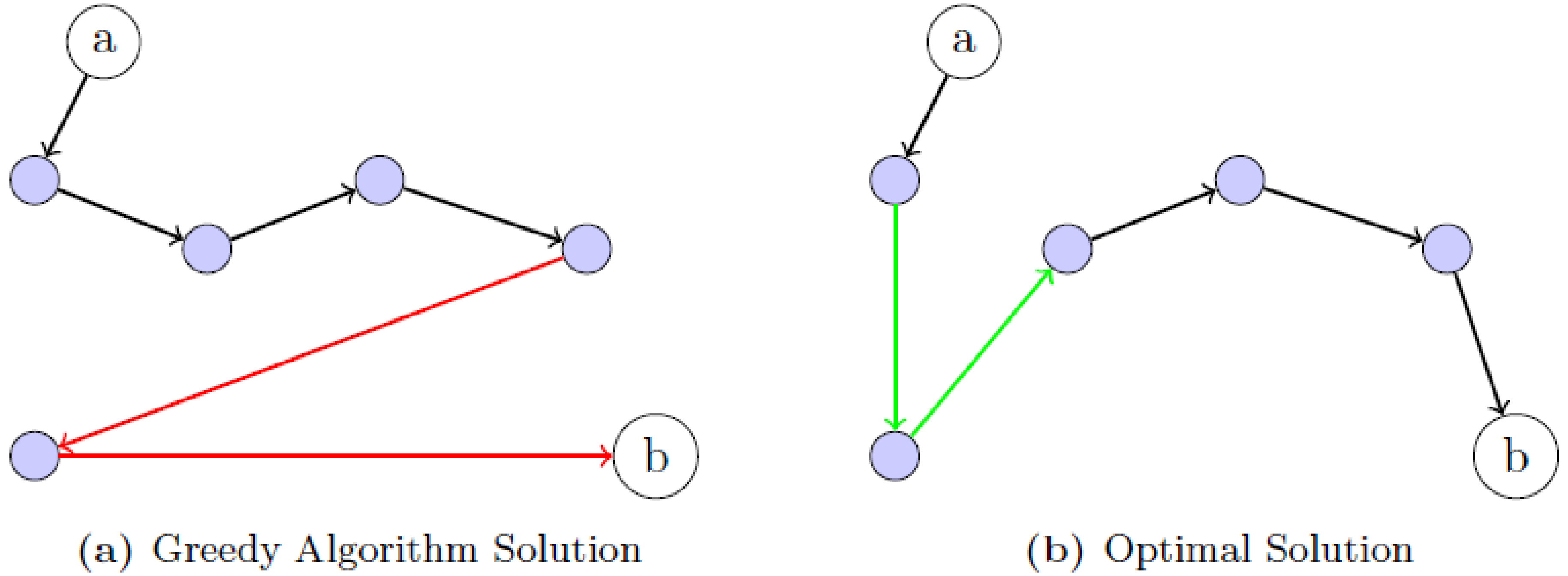
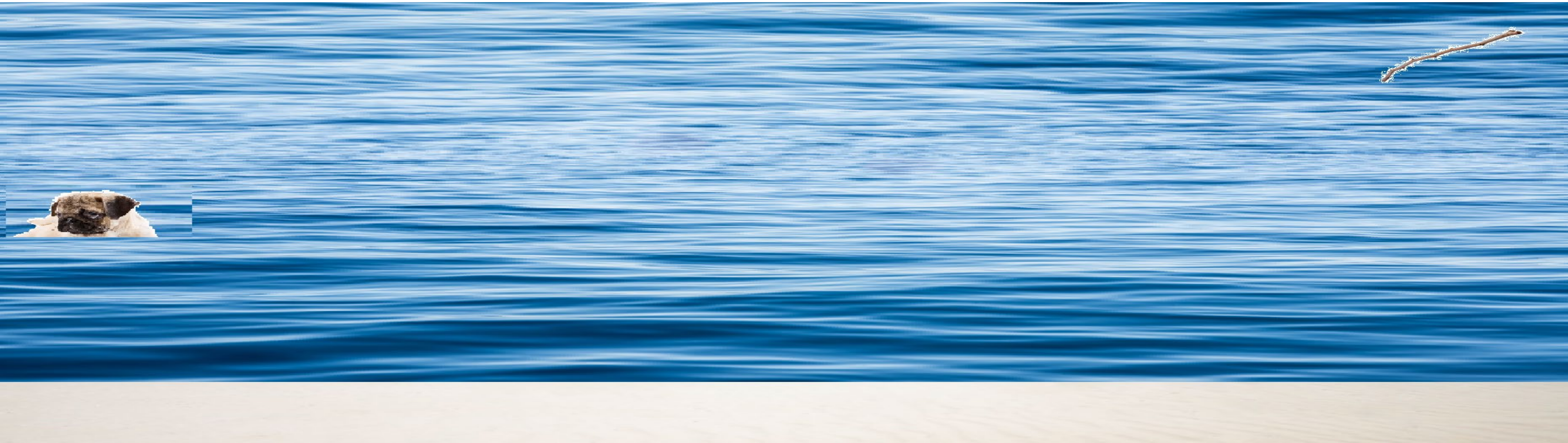


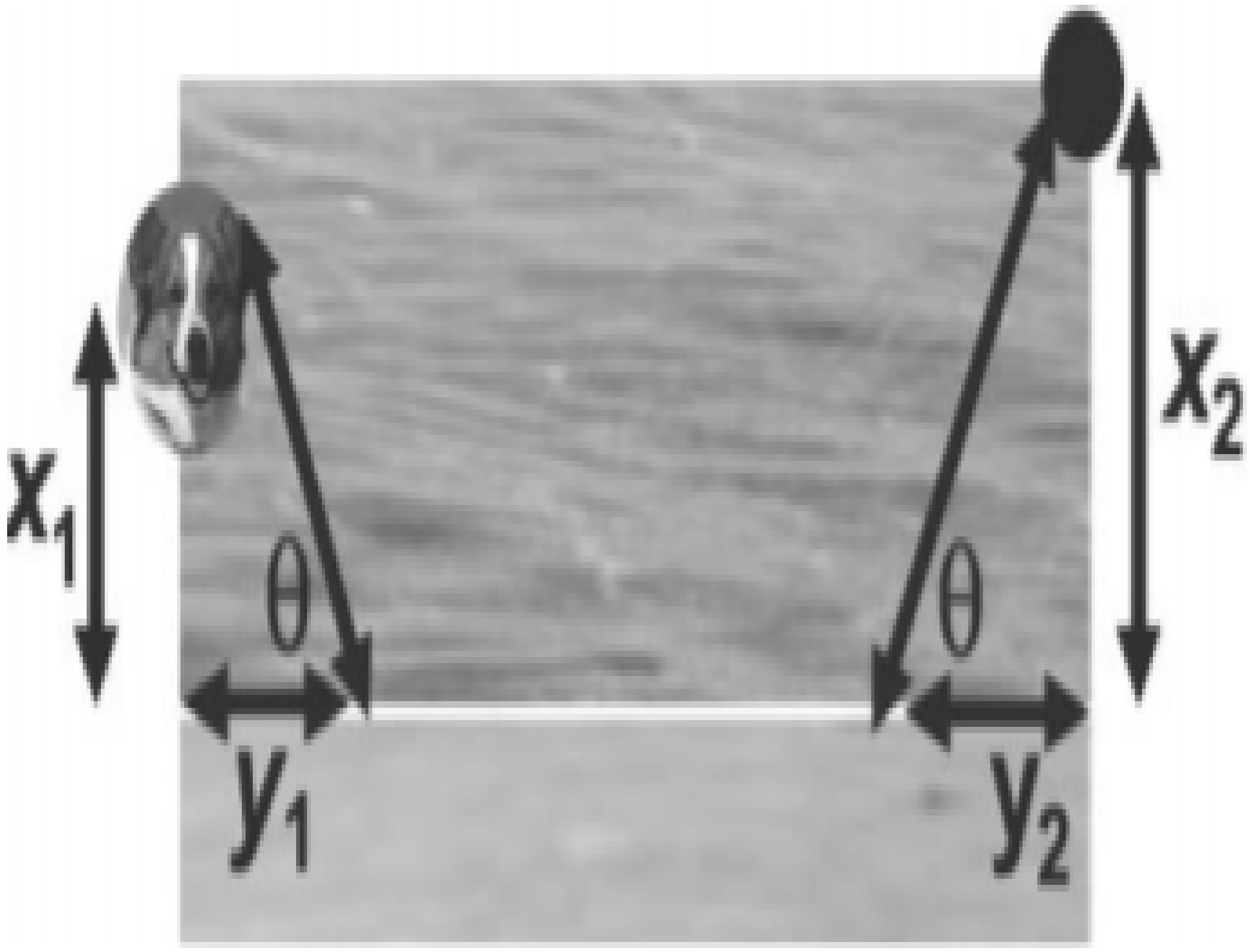
Figure 3. For this arrangement of cities, the greedy algorithm finds a sub-optimal path from city a to city b.

Greedy Algorithm issues….

- Timothy J. Pennings: Do dogs know calculus? http://www.math.pitt.edu/~bard/bardware/classes/0220/dkc.pdf

- Roland Minton and Timothy J. Pennings: Do dogs know bifurcations? https://www.maa.org/sites/default/files/pdf/upload_library/22/Polya/minton356.pdf





The physical problem also helps us determine the length of the top of this trapezoid. As $\frac{r}{s} \to \infty$, the running time along the beach approaches zero, so the total SRS time equals the time to swim $x_1 + x_2$ meters. At the bifurcation point, the S and SRS times are equal, so the S path must also have length $x_1 + x_2$ meters, as in Figure 5. Check this out geometrically!

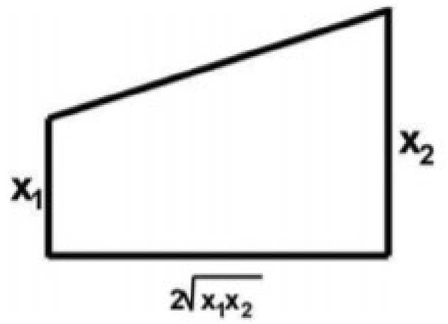


Figure 4. The SRS path.

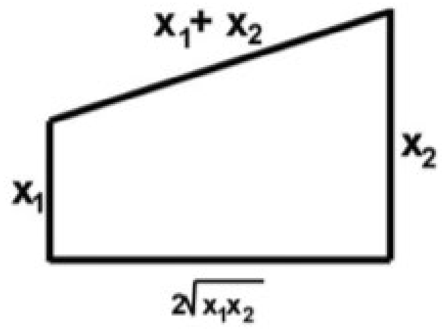


Figure 5. A mean triangle.