Intelligent Computing: Time to Gather Stones (a Tutorial)

Vladik Kreinovich

Department of Computer Science
University of Texas at El Paso
El Paso, Texas 79968, USA,
vladik@utep.edu
http://www.cs.utep.edu/vladik
http://www.cs.utep.edu/vladik/cs5354.19

1. Main Objective

- The main objective of this tutorial is to describe theoretical foundations for modern intelligent techniques.
- The emphasis will be on:
  - foundations of fuzzy techniques,
  - foundations of neural networks (in particular, deep neural networks), and
  - foundations of quantum computing.

2. Time to Gather Stones

- Many heuristic methods have been developed in intelligent computing.
- Some of them work well, some don't work so well.
- And promising techniques – that work well – often benefit from trial-and-error tuning.
- It is great to know and use all these techniques.
- It is also time to analyze why some technique work well and some don't.
- Following the Biblical analogy, we have gone through the time when we cast away stones in all directions.
- It is now time to gather stones, time to try to find the common patterns behind the successful ideas.
- Hopefully, in the future, this analysis will help.

3. Case Studies

- In this tutorial, we will concentrate on three classes of empirically successful semi-heuristic methods.
- Fuzzy techniques, techniques for translating:
  - expert knowledge described in terms of imprecise ("fuzzy") natural-language words like "small"
  - into precise numerical strategies.
- Neural networks (in particular, deep neural networks), techniques for learning a dependence from examples.
- Quantum computing, techniques that use quantum effects to make computations faster and more reliable.

Part I
Fuzzy Case

4. Fuzzy Techniques Are Needed

- In many application areas, we have experts whose experience we would like to capture.
- Often, experts' rules use imprecise ("fuzzy") words from natural language, like "small", "large", etc.
- To formalize these rules, L. Zadeh proposed special *fuzzy techniques.*
- A usual application of fuzzy techniques consists of the following three stages:
  1) reformulate expert knowledge in computer understandable terms – i.e., as numbers;
  2) process these numbers to come up with the degrees to which different actions are reasonable;
  3) if needed, "defuzzify" this "fuzzy" recommendation into an exact strategy.

5. First Stage of Fuzzy Technique

- In the first stage, we formalize the imprecise terms used by the experts, such as "small", "hot", and "fast".
- Each such term is described by assigning,
  - to different possible values x,
  - a degree $\mu(x)$ to which x satisfies this term (e.g., to which x is small).
- Some values $\mu(x)$ are obtained by asking the expert.
- However, there are infinitely many real numbers x, and we can only ask a finite number of questions,
- Thus, we need to perform *interpolation* to estimate the degrees $\mu(x)$ for intermediate values x.
- The result $\mu(x)$ is called the *membership function.*

6. Second Stage of Fuzzy Techniques: "And"- and "Or"-Operations

- Many expert rules involve several conditions.
- *Example:* a doctor will prescribe a certain medicine if the fever is high *and* blood pressure is normal.
- To handle such rules, we need to be able to transform:
  - the degrees $a = d(A)$ and $b = d(B)$ of individual conditions A and B
  - into a degree of confidence in the composite statement $A \,\&\, B$.
- The corresponding estimate $f_\&(a, b)$ is known as an *"and"-operation*, or, alternatively, as a *t-norm*.
- Similarly, we need an *"or"-operation* $f_\vee(a, b)$ (*t-conorm*) and a *negation operation* $f_\neg(a)$.

7. Third Stage of Fuzzy Techniques: Defuzzification

- After performing the first two stages,
  - for the given input x and for all possible control values u,
  - we get a degree $\mu(u)$ to which this control value is reasonable to apply.
- Sometimes, we want to use this expert knowledge in an automated system.
- In this case, we need to transform this membership function $\mu(u)$ into a single value $\overline{u}$.

8. Versions of Fuzzy Techniques

- There are many different membership functions $\mu(x)$, "and"- and "or"-operations, and defuzzifications.
- In practice, a few choices are the most efficient:
  - *trapezoid* $\mu(x)$: start with 0, linearly got to 1, stay at 1, then linearly decrease to 0;
  - $f_\&(a,b)=\min(a,b)$ or $f_\&(a,b)=a\cdot b$;
  - $f_\vee(a,b)=\max(a,b)$ or $f_\vee(a,b)=a+b-a\cdot b$;
  - negation operation $f_\neg(a)=1-a$; and
  - *centroid defuzzification* $\overline{u}=\dfrac{\int u\cdot\mu(u)\,du}{\int \mu(u)\,du}$.
- Similarly, for interval-valued case, both lower and upper membership functions are usually trapezoidal.
- We show that all these choices can be explained by the use of the simplest (linear) interpolation.

9. Linear Interpolation Is the Simplest

- Interpolation means that we find a function that attains known values at given points.
- The simplest possible non-constant functions are linear functions.
- They are also the least sensitive to uncertainty in x.
- We want the vector $e \stackrel{\text{def}}{=} (e_1, \ldots, e_k)$ of values $e_i \stackrel{\text{def}}{=} f'(x_i)$ to be as close to the ideal point $(0, \ldots, 0)$ as possible.
- The distance between the vector e and the 0 point is equal to $\sqrt{e_1^2 + \ldots + e_k^2}$.
- Minimizing the distance is equivalent to minimizing its square $e_1^2 + \ldots + e_k^2 = (f'(x_1))^2 + \ldots + (f'(x_k))^2$.
- This is the usual *Least Squares* method.

10. Linear Interpolation (cont-d)

- In the continuous case, we get an integral $\int (f'(x))^2\,dx$.
- Minimizing this interval, we get $f''(x) = 0$, so $f(x)$ is linear.
- If we know that $y_1 = f(x_1)$ and $y_2 = f(x_2)$, then these two values uniquely determine a linear function:

$$f(x) = f(x_1) + \frac{y_2 - y_1}{x_2 - x_1} \cdot (x - x_1).$$

- We will show that this simplest (linear) interpolation explains all usual choices of fuzzy techniques.

11. Explaining Trapezoid Membership Functions

- For each property like "small":
  - first, there are some values which are definitely not small (e.g., negative ones),
  - then some values which are small to some extent;
  - then, we have an interval of values which are definitely small;
  - this is followed by values which are somewhat small;
  - finally, we get values which are absolutely not small.
- Let us denote the values ("thresholds") that separate these regions by t_1, t_2, t_3, and t_4.
- Then: $\mu(x) = 0$ for $x \le t_1$; $\mu(x) = 1$ for $t_2 \le x \le t_3$; and $\mu(x) = 0$ for $x \ge t_4$.
- Linear interpolation indeed leads to trapezoid functions.

12. Explaining $f_\&(a, b) = a \cdot b$

- If one of the component statements A is false, then the composite statement $A \,\&\, B$ is also false: $f_\&(0, b) = 0$.
- If A is absolutely true, then our belief in $A \,\&\, B$ is equivalent to our degree of belief in B: $f_\&(1, b) = b$.
- Let us fix b and consider a function $F_b(a) \stackrel{\text{def}}{=} f_\&(a, b)$ that maps a into the value $f_\&(a, b)$.
- We know that $F_b(0) = 0$ and $F_b(1) = b$.
- Linear interpolation leads to $F_b(a) = a \cdot b$, i.e., to the algebraic product $f_\&(a, b) = a \cdot b$.
- Please note that:
  - while the resulting operation is commutative and associative,
  - we did not require commutativity or associativity;
  - all we required was linear interpolation.

13. What If We Additionally Require That $A \,\&\, A$ is Equivalent to A

- Another intuitive property of "and" is that for every B, "B and B" means the same as B: $f_\&(b, b) = b$.
- We know that $F_b(0) = f_\&(0, b) = 0$ and that $F_b(b) = f_\&(b, b) = b$.
- Thus, on the interval $[0, b]$, linear interpolation leads to $F_b(a) = a$, i.e., to $f_\&(a, b) = a$.
- From $F_b(b) = b$ and $F_b(1) = f_\&(1, b) = b$, we conclude that $f_\&(a, b) = F_b(a) = b$ for all $a \in [b, 1]$; so:
    - $f_\&(a, b) = a$ when $a \le b$ and
    - $f_\&(a, b) = b$ when $b \le a$.
- Thus, $f_\&(a, b) = \min(a, b)$.

14. Linear Interpolation Explains the Usual Choice of t-Conorms

- If A is absolutely true, then $A \vee B$ is also absolutely true: $f_\vee(a, b) = f_\vee(1, b) = 1$.
- If A is absolutely false, then our belief in $A \vee B$ is equivalent to our degree of belief in B: $f_\vee(0, b) = b$.
- For $G_b(a) \stackrel{\text{def}}{=} f_\vee(a, b)$, we get $G_b(0) = b$ and $G_b(1) = 1$.
- Linear interpolation leads to $G_b(a) = b + a \cdot (1 - b)$, i.e., to the algebraic sum $f_\vee(a, b) = a + b - a \cdot b$.
- Note that:
  - while the resulting operation is commutative and associative,
  - we did not require commutativity or associativity,
  - all we required was linear interpolation.

15. What If We Additionally Require That $A \vee A$ is Equivalent to A

- Another intuitive property of "or" is that for every B, "B or B" means the same as B: $f_\vee(b, b) = b$.
- We know that $G_b(0) = f_\vee(0, b) = b$ and that $G_b(b) = f_\vee(b, b) = b$.
- Thus, for $a \in [0, b]$, linear interpolation leads to $G_b(a) = b$, i.e., to $f_\&(a, b) = b$.
- From $G_b(b) = b$ and $G_b(1) = f_\vee(1, b) = 1$, we conclude that $f_\&(a, b) = G_b(a) = a$ for all $a \in [b, 1]$; so:
  - $f_\vee(a, b) = b$ when $a \le b$ and
  - $f_\vee(a, b) = a$ when $b \le a$.
- Thus, $f_\vee(a, b) = \max(a, b)$.

16. Simple Linear Interpolation Explains the Usual Choice of Negation Operations

- For the 2-valued logic, with truth values 1 ("true") and 0 ("false"), the negation operation is easy:
  - the negation of "false" is "true": $f_\neg(0) = 1$, and
  - the negation of "true" is "false": $f_\neg(1) = 0$.
- We want to extend this operation from the 2-valued set $\{0, 1\}$ to the whole interval $[0, 1]$.
- Linear interpolation leads to $f_\neg(a) = 1 - a$.
- This is exactly the most frequently used negation operation in fuzzy logic.

17. Simple Linear Interpolation Explains the Usual Choice of Defuzzification

- The desired control $\overline{u}$ should be close to reasonable control values u: $\overline{u} \approx u$.
- We have different possible control values u.
- Let us start with a simplified situation in which we have finitely many equally values $u_1, \ldots, u_k$.
- In this case, we want to find the values $\overline{u}$ for which $\overline{u} \approx u_1$, $\overline{u} \approx u_2$, $\ldots$, $\overline{u} \approx u_k$.
- Since the values u_i are different, we cannot get the exact equality in all k cases: $e_k \stackrel{\text{def}}{=} \overline{u} - u_k \neq 0$.
- We want the vector $e \stackrel{\text{def}}{=} (e_1, \ldots, e_k)$ to be as close to the ideal point $(0, \ldots, 0)$ as possible.
- The distance between the vector e and the 0 point is equal to $\sqrt{e_1^2 + \ldots + e_k^2}$.

18. Defuzzification (cont-d)

- Minimizing the distance is equivalent to minimizing its square $e_1^2 + \ldots + e_k^2 = (\overline{u} - u_1)^2 + \ldots + (\overline{u} - u_k)^2$.
- This is the usual *Least Squares* method.
- In the continuous case, we get an integral $\int (\overline{u} - u)^2 \, du$.
- This method works well if all the values u are equally possible.
- In reality, different values u have different degrees of possibility $\mu(u)$.
- If u is fully possible ($\mu(u) = 1$), we should keep the term $(\overline{u} - u)^2$ in the sum.
- If u if completely impossible ($\mu(u) = 0$), we should not consider this term at all.

19. Defuzzification: Result

- In general:
  - instead of simply adding the squares,
  - we first multiply each square by a weight $w(\mu(u))$ depending on $\mu(u)$, so that $w(1) = 1$ and $w(0) = 0$.
- Thus, we minimize $\int w(\mu(u)) \cdot (\overline{u} - u)^2 \, du$.
- Linear interpolation leads to $w(\mu) = \mu$, so we minimize

$$\int \mu(u) \cdot (\overline{u} - u)^2 \, du.$$

- Differentiating this expression with respect to $\overline{u}$ and equating the derivative to 0, we conclude that

$$\overline{u} = \frac{\int u \cdot \mu(u) \, du}{\int \mu(u) \, du}.$$

- So, simple linear interpolation explains the usual choice of centroid defuzzification.

20. Fuzzy Part: Conclusion

- In many real-life situations, we need to process expert knowledge.
- Experts often describe their knowledge by using imprecise ("fuzzy") terms from natural language.
- For processing such knowledge, Zadeh invented fuzzy techniques.
- Most efficient practical applications of fuzzy techniques use a specific combination of fuzzy techniques:
  - triangular or trapezoid membership functions,
  - simple t-norms (min or product),
  - simple t-conorms (max or algebraic sum), and
  - centroid defuzzification.
- For each of these choices, there exists an explanation of why this particular choice is efficient.

21. Conclusion (cont-d)

- Most efficient applications of fuzzy techniques use:
  - triangular or trapezoid membership functions,
  - simple t-norms (min or product),
  - simple t-conorms (max or algebraic sum), and
  - centroid defuzzification.
- For each of these choices, there exists an explanation of why this particular choice is efficient.
- The usual explanations, however, are different for different techniques.
- We show that all these choices can be explained by the use of the simplest (linear) interpolation.
- In our opinion, such a unform explanation makes the resulting choices easier to accept (and easier to teach).

Part II
Neural Network Case

22. Why Traditional Neural Networks: (Sanitized) History

- How do we make computers think?
- To make machines that fly it is reasonable to look at the creatures that know how to fly: the birds.
- To make computers think, it is reasonable to analyze how we humans think.
- On the biological level, our brain processes information via special cells called]it neurons.
- Somewhat surprisingly, in the brain, signals are electric – just as in the computer.
- The main difference is that in a neural network, signals are sequence of identical pulses.

23. Why Traditional NN: (Sanitized) History

- The intensity of a signal is described by the frequency of pulses.
- A neuron has many inputs (up to 10^4).
- All the inputs $x_1, \ldots, x_n$ are combined, with some loss, into a frequency $\sum\limits_{i=1}^{n} w_i \cdot x_i$.
- Low inputs do not active the neuron at all, high inputs lead to largest activation.
- The output signal is a non-linear function
$$y = f\left(\sum_{i=1}^{n} w_i \cdot x_i - w_0\right).$$
- In biological neurons, $f(x) = 1/(1 + \exp(-x))$.
- Traditional neural networks emulate such biological neurons.

24. Why Traditional Neural Networks: Real History

- At first, researchers ignored non-linearity and only used linear neurons.
- They got good results and made many promises.
- The euphoria ended in the 1960s when MIT's Marvin Minsky and Seymour Papert published a book.
- Their main result was that a composition of linear functions is linear (I am not kidding).
- This ended the hopes of original schemes.
- For some time, neural networks became a bad word.
- Then, smart researchers came us with a genius idea: let's make neurons non-linear.
- This revived the field.

25. Traditional Neural Networks: Main Motivation

- One of the main motivations for neural networks was that computers were slow.
- Although human neurons are much slower than CPU, the human processing was often faster.
- So, the main motivation was to make data processing faster.
- The idea was that:
  - since we are the result of billion years of ever improving evolution,
  - our biological mechanics should be optimal (or close to optimal).

26. How the Need for Fast Computation Leads to Traditional Neural Networks

- To make processing faster, we need to have many fast processing units working in parallel.
- The fewer layers, the smaller overall processing time.
- In nature, there are many fast linear processes – e.g., combining electric signals.
- As a result, linear processing (L) is faster than non-linear one.
- For non-linear processing, the more inputs, the longer it takes.
- So, the fastest non-linear processing (NL) units process just one input.
- It turns out that two layers are not enough to approximate any function.

27. Why One or Two Layers Are Not Enough

- With 1 linear (L) layer, we only get linear functions.
- With one nonlinear (NL) layer, we only get functions of one variable.
- With L→NL layers, we get $g\left(\sum_{i=1}^{n} w_i \cdot x_i - w_0\right)$.
- For these functions, the level sets $f(x_1, \ldots, x_n) = \text{const}$ are planes $\sum_{i=1}^{n} w_i \cdot x_i = c$.
- Thus, they cannot approximate, e.g., $f(x_1, x_2) = x_1 \cdot x_2$ for which the level set is a hyperbola.
- For NL→L layers, we get $f(x_1, \ldots, x_n) = \sum_{i=1}^{n} f_i(x_i)$.
- For all these functions, $d \stackrel{\text{def}}{=} \dfrac{\partial^2 f}{\partial x_1 \partial x_2} = 0$, so we also cannot approximate $f(x_1, x_2) = x_1 \cdot x_2$ with $d = 1 \neq 0$.

28. Why Three Layers Are Sufficient: Newton's Prism and Fourier Transform

- In principle, we can have two 3-layer configurations: L→NL→L and NL→L→NL.
- Since L is faster than NL, the fastest is L→NL→L:

$$y = \sum_{k=1}^{K} W_k \cdot f_k\left(\sum_{i=1}^{n} w_{ki} \cdot x_i - w_{k0}\right) - W_0.$$

- Newton showed that a prism decomposes while light (or any light) into elementary colors.
- In precise terms, elementary colors are sinusoids

$$A \cdot \sin(w \cdot t) + B \cdot \cos(w \cdot t).$$

- Thus, every function can be approximated, with any accuracy, as a linear combination of sinusoids:

$$f(x_1) \approx \sum_k (A_k \cdot \sin(w_k \cdot x_1) + B_k \cdot \cos(w_k \cdot x_1)).$$

29. Why Three Layers Are Sufficient (cont-d)

- Newton's prism result:

$$f(x_1) \approx \sum_k (A_k \cdot \sin(w_k \cdot x_1) + B_k \cdot \cos(w_k \cdot x_1)).$$

- This result was theoretically proven later by Fourier.
- For $f(x_1, x_2)$, we get a similar expression for each x_2, with $A_k(x_2)$ and $B_k(x_2)$.
- We can similarly represent $A_k(x_2)$ and $B_k(x_2)$, thus getting products of sines, and it is known that, e.g.:

$$\cos(a) \cdot \cos(b) = \frac{1}{2} \cdot (\cos(a+b) + \cos(a-b)).$$

- Thus, we get an approximation of the desired form with $f_k = \sin$ or $f_k = \cos$:

$$y = \sum_{k=1}^{K} W_k \cdot f_k \left(\sum_{i=1}^{n} w_{ki} \cdot x_i - w_{k0} \right).$$

30. Which Activation Functions $f_k(z)$ Should We Choose

- A general 3-layer NN has the form:

$$y = \sum_{k=1}^{K} W_k \cdot f_k\left(\sum_{i=1}^{n} w_{ki} \cdot x_i - w_{k0}\right) - W_0.$$

- Biological neurons use $f(z) = 1/(1+\exp(-z))$, but shall we simulate it?
- Simulations are not always efficient.
- E.g., airplanes have wings like birds but they do not flap them.
- Let us analyze this problem theoretically.
- There is always some noise c in the communication channel.
- So, we can consider either the original signals x_i or denoised ones $x_i - c$.

31. Which $f_k(z)$ Should We Choose (cont-d)

- The results should not change if we perform a full or partial denoising $z \to z' = z - c$.
- Denoising means replacing $y = f(z)$ with $y' = f(z-c)$.
- So, $f(z)$ should not change under shift $z \to z - c$.
- Of course, $f(z)$ cannot remain the same: if $f(z) = f(z-c)$ for all c, then $f(z) = \text{const}$.
- The idea is that once we re-scale x, we should get the same formula after we apply a natural y-re-scaling T_c:

$$f(x-c) = T_c(f(x)).$$

- Linear re-scalings are natural: they corresponding to changing units and starting points (like C to F).

32. Which Transformations Are Natural?

- An inverse T_c^{-1} to a natural re-scaling T_c should also be natural.
- A composition $y \to T_c(T_{c'}(y))$ of two natural re-scalings T_c and $T_{c'}$ should also be natural.
- In mathematical terms, natural re-scalings form a *group*.
- For practical purposes, we should only consider re-scaling determined by finitely many parameters.
- So, we look for a finite-parametric group containing all linear transformations.

33. A Somewhat Unexpected Approach

- N. Wiener, in *Cybernetics*, notices that when we approach an object, we have distinct phases:
  - first, we see a blob (the image is invariant under all transformations);
  - then, we start distinguishing angles from smooth but not sizes (projective transformations);
  - after that, we detect parallel lines (affine transformations);
  - then, we detect relative sizes (similarities);
  - finally, we see the exact shapes and sizes.
- Are there other transformation groups?
- Wiener argued: if there are other groups, after billions years of evolutions, we would use them.
- So he conjectured that there are no other groups.

34. Wiener Was Right

- Wiener's conjecture was indeed proven in the 1960s.
- In 1-D case, this means that all our transformations are fractionally linear:
$$f(z-c)=\frac{A(c)\cdot f(z)+B(c)}{C(c)\cdot f(z)+D(c)}.$$
- For $c=0$, we get $A(0)=D(0)=1$, $B(0)=C(0)=0$.
- Differentiating the above equation by c and taking $c=0$, we get a differential equation for $f(z)$:
$$-\frac{df}{dz}=(A'(0)\cdot f(z)+B'(0))-f(z)\cdot(C'(0)\cdot f(z)+D'(0)).$$
- So, $\dfrac{df}{C'(0)\cdot f^2+(A'(0)-C'(0))\cdot f+B'(0)}=-dz.$
- Integrating, we indeed get $f(z)=1/(1+\exp(-z))$ (after an appropriate linear re-scaling of z and $f(z)$).

35. How to Train Traditional Neural Networks: Main Idea

- *Reminder:* a 3-layer neural network has the form:

$$y = \sum_{k=1}^{K} W_k \cdot f\left(\sum_{i=1}^{n} w_{ki} \cdot x_i - w_{k0}\right) - W_0.$$

- We need to find the weights that best described observations $\left(x_1^{(p)}, \ldots, x_n^{(p)}, y^{(p)}\right)$, $1 \le p \le P$.
- We find the weights that minimize the mean square approximation error $E \stackrel{\text{def}}{=} \sum\limits_{p=1}^{P} \left(y^{(p)} - y_{NN}^{(p)}\right)^2$, where

$$y^{(p)} = \sum_{k=1}^{K} W_k \cdot f\left(\sum_{i=1}^{n} w_{ki} \cdot x_i^{(p)} - w_{k0}\right) - W_0.$$

- The simplest minimization algorithm is gradient descent: $w_i \to w_i - \lambda \cdot \dfrac{\partial E}{\partial w_i}$.

36. Towards Faster Differentiation

- To achieve high accuracy, we need many neurons.
- Thus, we need to find many weights.
- To apply gradient descent, we need to compute all partial derivatives $\dfrac{\partial E}{\partial w_i}$.
- Differentiating a function f is easy:
  - the expression f is a sequence of elementary steps,
  - so we take into account that $(f \pm g)' = f' \pm g'$, $(f \cdot g)' = f' \cdot g + f \cdot g'$, $(f(g))' = f'(g) \cdot g'$, etc.
- For a function that takes T steps to compute, computing f' thus takes $c_0 \cdot T$ steps, with $c_0 \le 3$.
- However, for a function of n variables, we need to compute n derivatives.
- This would take time $n \cdot c_0 \cdot T \gg T$: this is too long.

37. Faster Differentiation: Backpropagation

- Idea:
  - instead of starting from the variables,
  - start from the last step, and compute $\dfrac{\partial E}{\partial v}$ for all intermediate results v.
- For example, if the very last step is $E = a \cdot b$, then $\dfrac{\partial E}{\partial a} = b$ and $\dfrac{\partial E}{\partial b} = a$.
- At each step y, if we know $\dfrac{\partial E}{\partial v}$ and $v = a \cdot b$, then $\dfrac{\partial E}{\partial a} = \dfrac{\partial E}{\partial v} \cdot b$ and $\dfrac{\partial E}{\partial b} = \dfrac{\partial E}{\partial v} \cdot a$.
- At the end, we get all n derivatives $\dfrac{\partial E}{\partial w_i}$ in time
$$c_0 \cdot T \ll c_0 \cdot T \cdot n.$$
- This is known as *backpropagation*.

38. Beyond Traditional NN

- Nowadays, computer speed is no longer a big problem.
- What *is* a problem is *accuracy:* even after thousands of iterations, the NNs do not learn well.
- So, instead of computation speed, we would like to maximize learning accuracy.
- We can still consider L and NL elements.
- For the same number of variables w_i, we want to get more accurate approximations.
- For given number of variables, and given accuracy, we get N possible combinations.
- If all combinations correspond to different functions, we can implement N functions.
- However, if some combinations lead to the same function, we implement fewer different functions.

39. From Traditional NN to Deep Learning

- For a traditional NN with K neurons, each of $K!$ permutations of neurons retains the resulting function.
- Thus, instead of N functions, we only implement
$$\frac{N}{K!} \ll N \text{ functions.}$$
- Thus, to increase accuracy, we need to minimize the number K of neurons in each layer.
- To get a good accuracy, we need many parameters, thus many neurons.
- Since each layer is small, we thus need many layers.
- This is the *main idea* behind *deep learning*.

40. Empirical Formulas Behind Deep Learning Successes and How They Can Be Justified

- The general idea of deep learning is natural.
- However, any specific formulas that lead to deep learning successes are purely empirical.
- These formulas need to be explained.
- In this part of the tutorial:
  - we list such formulas, and
  - we briefly mention how the corresponding formulas can be explained.

41. Rectified Linear Neurons

- Traditional neural networks use complex nonlinear neurons.
- On contrast, deep networks utilize *rectified linear neurons* with the activation function

$$s_0(z) = \max(0, z).$$

- Our explanation is that:
  - this activation function is invariant under re-scaling (changing of the measuring unit) $z \to \lambda \cdot x$;
  - moreover, it is, in effect, the only activation function which is this invariant, and
  - it is the only activation f-n optimal with respect to any scale-invariant optimality criterion.

42. Combining Several Results

- To speed up the training, the current deep learning algorithms use dropout techniques:
  - they train several sub-networks on different portions of data, and then
  - "average" the results.
- A natural idea is to use arithmetic mean for this "averaging".
- However, empirically, geometric mean works much better.
- How to explain this empirical efficiency?
- It turns out that
  - this choice is scale-invariant – and,
  - in effect, it is the only scale-invariant choice.

43. Softmax

- In deep learning:
  - instead of selecting an alternative for which the objective function $f(x)$ is the largest possible,
  - we use so-called *softmax* – i.e., select each alternative x with probability proportional to $\exp(\alpha\cdot f(x))$.
- In general, we could select any increasing function $F(z)$ and select probabilities proportional to $F(f(x))$.
- So why exponential function is the most successful?

44. Softmax: Explanation

- When we use softmax, the probabilities do not change if we simply shift all the values $f(x)$.
- I.e., if we change them to $f(x) + c$ for some c.
- This shift does not change the original optimization problem.
- Moreover, exponential functions are the only ones which lead to such shift-invariant selection.
- The exponential functions are only ones which optimal under a shift-invariant optimality criterion.

45. Need for Convolutional Neural Networks

- In many practical situations, the available data comes:
  - in terms of *time series* – when we have values measured at equally spaced time moments – or
  - in terms of an *image* – when we have data corresponding to a grid of spatial locations.
- Neural networks for processing such data are known as *convolutional neural networks.*

46. Need for Pooling

- We want to decrease the distortions caused by measurement errors.
- For that, we take into account that usually, the actual values at nearby points in time or space are close to each other.
- As a result,
  - instead of using the measurement-distorted value at each point,
  - we can take into account that values at nearby points are close, and
  - combine ("pool together") these values into a single more accurate estimate.

47. Which Pooling Techniques Work Better: Empirical Results

- In principle, we can have many different pooling algorithms.
- It turns out that empirically, in general, the most efficient pooling algorithm is *max-pooling*:

$$a = \max(a_1, \ldots, a_m).$$

- The next efficient is *average pooling*, when we take the arithmetic average $a = \dfrac{a_1 + \ldots + a_m}{m}$.
- In this tutorial, we provide a theoretical explanation for this empirical observation.
- Namely, we prove that max and average poolings are indeed optimal.

Main Objective
Time to Gather Stones
Case Studies
Fuzzy Case
Neural Network Case
Quantum Computing
Proofs (if time allows)
Home Page
Title Page
◀◀ ▶▶
◀ ▶
Page 50 of 147
Go Back
Full Screen
Close
Quit

48. Pooling: Towards a Precise Definition

- Based on m values $a_1, \ldots, a_m$, we want to generate a single value a.
- In the case of arithmetic average, we select a for which $a_1 + \ldots + a_m = a + \ldots + a$ (m times).
- In general, pooling means that:
  - we select some combination operation $*$ and
  - we then select the value a for which $a_1 * \ldots * a_m = a * \ldots * a$ (m times).
- For example:
  - if, as a combination operation, we select $\max(a, b)$,
  - then the corresponding condition $\max(a_1, \ldots, a_n) = \max(a, \ldots, a) = a$ describes the max-pooling.
- From this viewpoint, selecting pooling means selecting an appropriate combination operation.

49. Natural Properties of a Combination Operation

- The combination operation transforms:
  - two non-negative values – such as intensity of an image at a given location
  - into a single non-negative value.
- The result of applying this operation should not depend on the order in which we combine the values.
- Thus, we should have $a * b = b * a$ (commutativity) and $a * (b * c) = (a * b) * c$ (associativity).

50. What Does It Mean to Have an Optimal Pooling?

- Optimality means that on the set of all possible combination operations, we have a preference relation $\preceq$.
- $A \preceq B$ means that the operation B is better than (or of the same quality as) the operation A.
- This relation should be transitive:
  - if C is better than B and B is better than A,
  - then C should be better than A.
- An operation A is optimal if it is better than (or of the same quality as) any other operation B: $B \preceq A$.
- For some preference relations, we may have several different optimal combination operations.
- We can then use this non-uniqueness to optimize something else.

51. What Is Optimal Pooling (cont-d)

- Example:
  - if there are several different combination operations with the best average-case accuracy,
  - we can select, among them, the one for which the average computation time is the smallest possible.
- If after this, we still get several optimal operations,
  - we can use the remaining non-uniqueness
  - to optimize yet another criterion.
- We do this until we get a *final* criterion, for which there is only one optimal combination operation.

52. Scale-Invariance

- Numerical values of a physical quantity depend on the choice of a measuring unit.
- For example, if we replace meters with centimeters, the numerical quantity is multiplied by 100.
- In general:
  - if we replace the original unit with a unit which is λ times smaller,
  - then all numerical values get multiplied by λ.
- It is reasonable to require that the preference relation should not change if we change the measuring unit.
- Let us describe this requirement in precise terms.

53. Scale-Invariance (cont-d)

- If, in the original units, we had the operation $a * b$, then, in the new units, the operation will be as follows:
  - first, we transform the value a and b into the new units, so we get $a' = \lambda \cdot a$ and $b' = \lambda \cdot b$;
  - then, we combine the new numerical values, getting $(\lambda \cdot a) * (\lambda \cdot b)$;
  - finally, we re-scale the result to the original units, getting $a R_\lambda(*) b \stackrel{\text{def}}{=} \lambda^{-1} \cdot ((\lambda \cdot a) * (\lambda \cdot b))$.
- It therefore makes sense to require that if $* \preceq *'$, then for every $\lambda > 0$, we get $R_\lambda(*) \preceq R_\lambda(*')$.

54. Shift-Invariance

- The numerical values also change if we change the starting point for measurements.
- For example, when measuring intensity:
  - we can measure the actual intensity of an image,
  - or we can take into account that there is always some noise $a_0 > 0$, and
  - use the noise-only level a_0 as the new starting point.
- In this case, instead of each original value a, we get a new numerical value $a' = a - a_0$.

55. Shift-Invariance (cont-d)

- If we apply the combination operation in the new units, then in the old units, we get a slightly different result:
  - first, we transform the value a and b into the new units, so we get $a' = a - a_0$ and $b' = b - a_0$;
  - then, we combine the new numerical values, getting

$$(a - a_0) * (b - a_0);$$

  - finally, we re-scale the result to the original units, getting $a S_{a_0}(*) b \stackrel{\text{def}}{=} (a - a_0) * (b - a_0) + a_0$.
- It makes sense to require that the preference relation not change if we simply change the starting point.
- So if $* \preceq *'$, then for every a_0, we get $S_{a_0}(*) \preceq S_{a_0}(*')$.

56. Weak Version of Shift-Invariance

- Alternatively, we can have a weaker version of this "shift-invariance".
- Namely, we require that shifts in a and b imply a possibly different shift in $a * b$, i.e.,
  - if we shift both a and b by a_0,
  - then the value $a * b$ is shifted by some value $f(a_0)$ which is, in general, different from a_0.
- Now, we are ready to formulation our results.

57. Definitions

- *By a* combination operation, *we mean a commutative, associative operation $a * b$ that:*
  - *transforms two non-negative real numbers a and b*
  - *into a non-negative real number $a * b$.*
- *By an* optimality criterion, *we need a transitive reflexive relation $\preceq$ on the set of all combination operations.*
- *We say that a combination operation $*_{\rm opt}$ is* optimal *w.r.t. $\preceq$ if $* \preceq *_{\rm opt}$ for all combination operations $*$.*
- *We say that $\preceq$ is* final *if there exists exactly one $\preceq$-optimal combination operation.*
- *We say that an optimality criterion is* scale-invariant *if for all $\lambda > 0$, $* \preceq *'$ implies $R_\lambda(*) \preceq R_\lambda(*')$, where:*

$$a R_\lambda(*) b \stackrel{\rm def}{=} \lambda^{-1} \cdot ((\lambda \cdot a) * (\lambda \cdot b)).$$

58. Definitions and First Result

- *We say that an optimality criterion is* shift-invariant *if for all a_0, $* \preceq *'$ implies $S_{a_0}(*) \preceq S_{a_0}(*')$, where:*

$$aS_{a_0}(*)b \stackrel{\text{def}}{=} ((a - a_0) * (b - a_0)) + a_0.$$

- *We say that $\preceq$ is* weakly shift-invariant *if for every a_0, there exists $f(a_0)$ s.t. $* \preceq *'$ implies $W_{a_0}(*) \preceq W_{a_0}(*')$,*

$$\textit{where}\ \ aW_{a_0}(*)b \stackrel{\text{def}}{=} ((a - a_0) * (b - a_0)) + f(a_0).$$

- **Proposition 1.** *For every final, scale- and shift-invariant $\preceq$, the optimal combination operation $*$ is*

$$a * b = \min(a, b)\ \ \textit{or}\ a * b = \max(a, b).$$

- *This result explains why max-pooling is empirically the best combination operation.*

- Note that this result does not contradict uniqueness as we requested.

59. Results (cont-d)

- Indeed, there are several different final scale- and shift-invariant optimality criteria.
- For each of these criteria, there is only one optimal combination operation.
- For some of these optimality criteria, the optimal combination operation is $\min(a, b)$.
- For other criteria, the optimal combination operation is $\max(a, b)$.
- **Proposition 2.** *For every final, scale-invariant and weakly shift-invariant $\preceq$, the optimal $*$ is:*

$$a * b = 0, \quad a * b = \min(a, b), \quad a * b = \max(a, b), \ \textit{or}$$

$$a * b = a + b.$$

- *This result explains why max-pooling and average-pooling are empirically the best combination operations.*

Part III
Quantum Computing

60. Why Quantum Computing

- In many practical problems, we need to process large amounts of data in a limited time.
- To be able to do it, we need computations to be as fast as possible.
- Computations are already fast.
- However, there are many important problems for which we still cannot get the results on time.
- For example, we can predict with a reasonable accuracy where the tornado will go in the next 15 minutes.
- However, these computations take days on the fastest existing high performance computer.
- One of the main limitations: the speed of all the processes is limited by the speed of light $c \approx 3 \cdot 10^5$ km/sec.

61. Why Quantum Computing (cont-d)

- For a laptop of size ≈ 30 cm, the fastest we can send a signal across the laptop is $\dfrac{30\text{ cm}}{3 \cdot 10^5\text{ km/sec}} \approx 10^{-9}$ sec.
- During this time, a usual few-Gigaflop laptop performs quite a few operations.
- To further speed up computations, we thus need to further decrease the size of the processors.
- We need to fit Gigabytes of data – i.e., billions of cells – within a small area.
- So, we need to attain a very small cell size.
- At present, a typical cell consists of several dozen molecules.
- As we decrease the size further, we get to a few-molecule size.

62. Why Quantum Computing (cont-d)

- At this size, physics is different: quantum effects become dominant.
- At first, quantum effects were mainly viewed as a nuisance.
- For example, one of the features of quantum world is that its results are usually probabilistic.
- So, if we simply decrease the cell size but use the same computer engineering techniques, then:
  - instead of getting the desired results all the time,
  - we will start getting other results with some probability.
- This probability of undesired results increases as we decrease the size of the computing cells.

63. Why Quantum Computing (cont-d)

- However, researchers found out that:
  - by appropriately modifying the corresponding algorithms,
  - we can avoid the probability-related problem and, even better, make computations faster.
- The resulting algorithms are known as algorithms of *quantum computing*.

64. Lemon into Lemonade

- In non-quantum computing, finding an element in an unsorted database with n entries may require time n.
- Indeed, we may need to look at each record.
- In quantum computing, it is possible to find this element in much smaller time $\sqrt{n}$.

65. Quantum Computing Will Enable Us to Decode All Traditionally Encoded Messages

- One of the spectacular algorithms of quantum computing is Shor's algorithm for fast factorization.
- Most encryption schemes – the backbone of online commerce – are based on the RSA algorithm.
- This algorithm is based on the difficulty of factorizing large integers.
- To form an at-present-unbreakable code, the user selects two large prime numbers P_1 and P_2.
- These numbers form his private code.
- He then transmits to everyone their product $n = P_1 \cdot P_2$ that everyone can use to encrypt their messages.
- At present, the only way to decode this message is to know the values P_i.

66. Quantum Computing Can Decode All Traditionally Encoded Messages (cont-d)

- Shor's algorithm allows quantum computers to effectively find P_i based on n.
- Thus, it can read practically all the secret messages that have been sent so far.
- This is one governments invest in the design of quantum computers.

67. Quantum Cryptography: an Unbreakable Alternative to the Current Cryptographic Schemes

- That RSA-based cryptographic schemes can be broken by quantum computing.
- However, this does not mean that there will be no secrets.
- Researchers have invented a quantum-based encryption scheme that cannot be thus broken.
- This scheme, by the way, is already used for secret communications.

68. Remaining Problems And What We Do in This Tutorial

- In addition to the current cryptographic scheme, one can propose its modifications.
- This possibility raises a natural question: which of these scheme is the best?
- In this tutorial, we show that the current cryptographic scheme is, in some reasonable sense, optimal.

69. Quantum Physics: Possible States

- One of the main ideas behind quantum physics is that in the quantum world,
  - in addition to the regular states,
  - we can also have linear combinations of these states, with complex coefficients.
- Such combinations are known as *superpositions*.
- A single 1-bit memory cell in the classical physics can only have states 0 and 1.
- In quantum physics, these states are denoted by $|0\rangle$ and $|1\rangle$.
- We can also have superpositions $c_0 \cdot |0\rangle + c_1 \cdot |1\rangle$, where c_0 and c_1 are complex numbers.

70. Measurements in Quantum Physics

- What will happen if we try to measure the bit in the superposition state $c_0 \cdot |0\rangle + c_1 \cdot |1\rangle$?
- According to quantum physics, as a result of this measurement, we get:
  - 0 with probability $|c_0|^2$ and
  - 1 with probability $|c_1|^2$.
- After the measurement, the state also changes:
  - if the measurement result is 0, the state will turn into $|0\rangle$, and
  - if the measurement result is 1, the state will turn into $|1\rangle$.

71. Measurements in Quantum Physics (cont-d)

- Since we can get either 0 or 1, the corresponding probabilities should add up to 1; so:
  - for the expression $c_0 \cdot |0\rangle + c_1 \cdot |1\rangle$ to represent a physically meaningful state,
  - the coefficients c_0 and c_1 must satisfy the condition

$$|c_0|^2 + |c_1|^2 = 1.$$

72. Operations on Quantum States

- We can perform *unitary* operations, i.e., linear transformations that preserve the property

$$|c_0|^2 + |c_1|^2 = 1.$$

- A simple example of a unary transformation is *Walsh-Hadamard (WH)* transformation:

$$|0\rangle \to |0'\rangle \stackrel{\text{def}}{=} \frac{1}{\sqrt{2}} \cdot |0\rangle + \frac{1}{\sqrt{2}} \cdot |1\rangle;$$

$$|1\rangle \to |1'\rangle \stackrel{\text{def}}{=} \frac{1}{\sqrt{2}} \cdot |0\rangle - \frac{1}{\sqrt{2}} \cdot |1\rangle.$$

- What is the geometric meaning of this transformation?

73. Operations on Quantum States (cont-d)

- By linearity: $c'_0 \cdot |0'\rangle + c'_1 \cdot |1'\rangle =$
$$c'_0 \cdot \left(\frac{1}{\sqrt{2}} \cdot |0\rangle + \frac{1}{\sqrt{2}} \cdot |1\rangle\right) + c'_1 \cdot \left(\frac{1}{\sqrt{2}} \cdot |0\rangle - \frac{1}{\sqrt{2}} \cdot |1\rangle\right) =$$
$$\left(\frac{1}{\sqrt{2}} \cdot c'_0 + \frac{1}{\sqrt{2}} \cdot c'_1\right) \cdot |0\rangle + \left(\frac{1}{\sqrt{2}} \cdot c'_0 - \frac{1}{\sqrt{2}} \cdot c'_1\right) \cdot |1\rangle.$$
- Thus, $c'_0 \cdot |0'\rangle + c'_1 \cdot |1'\rangle = c_0 \cdot |0\rangle + c_1 \cdot |1\rangle$, where
$$c_0 = \frac{1}{\sqrt{2}} \cdot c'_0 + \frac{1}{\sqrt{2}} \cdot c'_1 \text{ and } c_1 = \frac{1}{\sqrt{2}} \cdot c'_0 - \frac{1}{\sqrt{2}} \cdot c'_1.$$
- Let us represent each of the two pairs (c_0, c_1) and (c'_0, c'_1) as a point in the 2-D plane (x, y).
- Then the above transformation resembles the formulas for a clockwise rotation by an angle θ:
$$x' = \cos(\theta) \cdot x + \sin(\theta) \cdot y;$$
$$y' = -\sin(\theta) \cdot x + \cos(\theta) \cdot y.$$

74. Operations on Quantum States (cont-d)

- Specifically, for $\theta = 45^\circ$, we have $\cos(\theta) = \sin(\theta) = \dfrac{1}{\sqrt{2}}$ and thus, the rotation takes the form

$$x' = \frac{1}{\sqrt{2}} \cdot x + \frac{1}{\sqrt{2}} \cdot y; \quad y' = -\frac{1}{\sqrt{2}} \cdot x + \frac{1}{\sqrt{2}} \cdot y.$$

- In these terms, can see that the WH transformation from (c_0', c_1') and (c_0, c_1) is:
  - a rotation by 45 degrees
  - followed by a reflection with respect to the x-axis: $(c_0, c_1) \to (c_0, -c_1)$.
- One can check that if we apply WH transformation twice, then we get the same state as before.

75. Operations on Quantum States (cont-d)

- Indeed, due to linearity,

$$\mathrm{WH}(0') = \mathrm{WH}\left(\frac{1}{\sqrt{2}} \cdot |0\rangle + \frac{1}{\sqrt{2}} \cdot |1\rangle\right) =$$

$$\frac{1}{\sqrt{2}} \cdot WH(|0\rangle) + \frac{1}{\sqrt{2}} \cdot WH(|1\rangle) =$$

$$\frac{1}{\sqrt{2}} \cdot \left(\frac{1}{\sqrt{2}} \cdot |0\rangle + \frac{1}{\sqrt{2}} \cdot |1\rangle\right) + \frac{1}{\sqrt{2}} \cdot \left(\frac{1}{\sqrt{2}} \cdot |0\rangle - \frac{1}{\sqrt{2}} \cdot |1\rangle\right) =$$

$$|0\rangle.$$

- Similarly, $\mathrm{WH}(|1'\rangle) = |1\rangle$.

76. Measurements of Quantum 1-Bit Systems

- According to quantum measurement:
  - if we measure the bit 0 or 1 in each of the states $|0'\rangle$ or $|1'\rangle$,
  - then we will get 0 or 1 with equal probability $1/2$.
- So, if we measure 0 or 1, then:
  - if we are in the state $|0\rangle$, then the state does not change and we get 0 with probability 1;
  - if we are in the state $|1\rangle$, then the state does not change and we get 1 with probability 1;
  - if we are in one of the states $|0'\rangle$ or $|1'\rangle$, then:
    - with probability $1/2$, we get the measurement result 0 and the state changes into $|0\rangle$; and
    - with probability $1/2$, we get the measurement result 1 and the state changes into $|1\rangle$.

77. Case of Quantum 1-Bit Systems (cont-d)

- We can also measure whether we have $|0'\rangle$ or $|1'\rangle$.
- In this case, similarly:
  - if we are in the state $|0'\rangle$, then the state does not change and we get $0'$ with probability 1;
  - if we are in the state $|1'\rangle$, then the state does not change and we get $1'$ with probability 1;
  - if we are in one of the states $|0\rangle$ or $|1\rangle$, then:
    * with probability $1/2$, we get the measurement result $0'$ and the state changes into $|0'\rangle$; and
    * with probability $1/2$, we get the measurement result $1'$ and the state changes into $|1'\rangle$.

78. Main Idea of Quantum Cryptography

- The sender – who, in cryptography, is usually called Alice – sends each bit
  - either as $|0\rangle$ or $|1\rangle$ (this orientation is usually denoted by $+$)
  - or as $|0'\rangle$ or $|1'\rangle$ (this orientation is usually denoted by $\times$).
- The receiver – who, in cryptography, is usually called Bob – tries to extract the information from the signal.
- Extracting numerical information from a physical object is nothing else but measurement.
- Thus, to extract the information from Alice's signal, Bob needs to perform some measurement.
- Since Alice uses one of the two orientations $+$ or $\times$, it is reasonable for Bob to also use one of these orientations.

79. Sender and Receiver Must Use the Same Orientation

- If for some bit:
  - Alice and Bob use the same orientation,
  - then Bob will get the exact same signal that Alice has sent.
- The situation is completely different if Alice and Bob use different orientations.
- For example, assume that:
  - Alice sends a 0 bit in the $\times$ orientation, i.e., sends the state $|0'\rangle$, and
  - Bob uses the $+$ orientation to measure the signal.

80. We Need Same Orientation (cont-d)

- For the state $|0'\rangle = \frac{1}{\sqrt{2}} \cdot |0\rangle + \frac{1}{\sqrt{2}} \cdot |1\rangle$:
  - with probability $\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$, Bob will measure 0, and
  - with probability $\left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2}$, Bob will measure 1.
- The same results, with the same probabilities, will happen if Alice sends a 1 bit in the $\times$ orientation, i.e., $|1'\rangle$.
- Thus, by observing the measurement result, Bob will not be able to tell whether Alice send 0 or 1.
- The information will be lost.
- Similarly, the information will be lost if Alice uses a $+$ orientation and Bob uses a $\times$ orientation.

81. What If We Have an Eavesdropper?

- What if an eavesdropper – usually called Eve – gains access to the same communication channel?
- In non-quantum eavesdropping, Eve can measure each bit that Alice sends and thus, get the whole message.
- In non-quantum physics, measurement does not change the signal.
- Thus, Bob gets the same signal that Alice has sent.
- Neither Alice not Bob will know that somebody eavesdropped on their communication.
- In quantum physics, the situation is different.
- One of the main features of quantum physics is that measurement, in general, changes the signal.
- Eve does not know in which of the two orientations each bit is sent.

82. What If We Have an Eavesdropper (cont-d)

- So, she can select the wrong orientation for her measurement.
- As a result, e.g.,
  - if Alice and Bob agreed to use the $\times$ orientation for transmitting a certain bit,
  - but Eve selects a $+$ orientation,
  - then Eve's measurement will change Alice's signal – and Bob will only get the distorted message.
- For example, if Alice sent $|0'\rangle$, then:
  - after Eve's measurement,
  - the signal will become either $|0\rangle$ or $|1\rangle$, with probability $1/2$ of each of these options.

83. What If We Have an Eavesdropper (cont-d)

- In each of the options:
  - when Bob measures the resulting signal ($|0\rangle$ or $|1\rangle$) by using his agreed-upon $\times$ orientation ($|0'\rangle, |1'\rangle$),
  - Bob will get 0 or 1 with probability 1/2 – instead of the original signal that Alice has sent.

84. Quantum Cryptography Helps to Detect an Eavesdropper

- If there is an eavesdropper, then:
  - with certain probability,
  - the signal received by Bob will be different from what Alice sent.
- Thus, by comparing what Alice sent with what Bob received, we can see that something was interfering.
- Thus, we will be able to detect the presence of the eavesdropper.
- Let us describe how this idea is implemented in the current quantum cryptography algorithm.

85. Sending a Preliminary Message

- Before Alice sends the actual message, she needs to check that the communication channel is secure.
- For this purpose, Alice uses a random number generator to select n random bits $b_1, \ldots, b_n$.
- Each of them is equal to 0 or 1 with probability $1/2$.
- These bits will be sent to Bob.
- Alice also selects n more random bits $r_1, \ldots, r_n$.
- Based on these bits, Alice sends the bits b_i as follows:
  - if $r_i = 0$, then the bit b_i is sent in $+$ orientation, i.e., Alice sends $|0\rangle$ if $b_i = 0$ and $|1\rangle$ if $b_i = 1$;
  - if $r_i = 1$, then the bit b_i is sent in $\times$ orientation, i.e., Alice sends $|0'\rangle$ if $b_i = 0$ and $|1'\rangle$ if $b_i = 1$.

86. Receiving the Preliminary Message

- Independently, Bob selects n random bits $s_1, \ldots, s_n$.
- They determine how he measures the signal that he receives from Alice:
  - if $s_i = 0$, then Bob measures whether the i-th received signal is $|0\rangle$ or $|1\rangle$;
  - if $s_i = 1$, then Bob measures whether the i-th received signal is $|0'\rangle$ or $|1'\rangle$.

87. Checking for Eavesdroppers

- After this, for k out of n bits, Alice openly sends to Bob her bits b_i and her orientations r_i.
- Bob sends to Alice his orientations s_i and the signals b'_i that he measured.
- In half of the cases, the orientations r_i and s_i should coincide.
- In which case, if there is no eavesdropper,
  - the signal b'_i measured by Bob
  - should coincide with the signal b_i that Alice sent.
- So, if $b'_i \neq b_i$ for some i, this means that there is an eavesdropper.
- If there is an eavesdropper, then with probability $1/2$, Eve will select a different orientation.

88. Checking for Eavesdroppers (cont-d)

- In half of such cases, the eavesdropping with change the original signal.
- So, for each bit, the probability that we will have $b'_i \neq b_i$ is equal to $1/4$.
- Thus, the probability that the eavesdropper will not be detected by this bit is $1 - 1/4 = 3/4$.
- The probability that Eve will not be detected in all $k/2$ cases is the product $(3/4)^{k/2}$.
- For a sufficiently large k, this probability of not-detecting-eavesdropping is very small.
- Thus, if $b'_i = b_i$ for all k bits i, this means that with high confidence, there is no eavesdropping.
- So, the communication channel between Alice and Bob is secure.

89. Preparing to Send a Message

- Now, for each of the remaining $(n-k)$ bits, Alice and Bob openly exchange orientations r_i and s_i.
- For half of these bits, these orientations must coincide.
- For these bits, since there is no eavesdropping, Alice and Bob know that:
  - the signal b'_i measured by Bob
  - is the same as the signal b_i sent to Alice.
- So, there are $B \stackrel{\text{def}}{=} (n-k)/2$ bits $b_i = b'_i$ that they both know but no one else knows.

90. Sending and Receiving the Actual Message

- Now, Alice takes the B-bit message $m_1, \ldots, m_B$ that she wants to send.
- She forms the encoded message $m'_i \stackrel{\text{def}}{=} m_i \oplus b_i$, where $\oplus$ means addition modulo 2 (same as exclusive or).
- Alice openly sends the encoded message m'_i.
- Upon receiving the message m'_i, Bob reconstructs the original message as $m_i = m'_i \oplus b_i$.

91. A General Family of Quantum Cryptography Algorithms: Description

- In the current quantum cryptography algorithm, Alice selects $+$ and $\times$ with probability 0.5.
- Similarly, Bob selects one of the two possible orientations $+$ and $\times$ with probability 0.5.
- It is therefore reasonable to consider a more general scheme, in which:
  - Alice selects the orientation $+$ with some probability a_+ (which is not necessarily equal to 0.5), and
  - Bob select the orientation $+$ with some probability b_+ (which is not necessarily equal to 0.5).
- Which a_+ and b_+ should they choose to make the connection maximally secure?
- I.e., to maximize the probability of detecting the eavesdropper?

92. What Do We Want to Maximize?

- We want to maximize the probability of detecting an eavesdropper.
- The eavesdropper also selects one of the two orientations $+$ or $\times$.
- Let e_+ be the probability with which the eavesdropper (Eve) select the orientation $+$.
- Then Eve will select $\times$ with the remaining probability $e_\times = 1 - e_+$.
- We know that Alice and Bob can only use bits for which their selected orientations coincide.
- If Eve selects the same orientation, then her observation will also not change this bit.
- Thus, we will not be able to detect the eavesdropping.

93. What Do We Want to Maximize (cont-d)

- We can detect the eavesdropping only when A and B have the same orientation, but E has a different one.
- There are two such cases:
  - the first case is when Alice and Bob select $+$ and Eve selects $\times$;
  - the second case is when Alice and Bob select $\times$ and Eve selects $+$.
- Alice, Bob, and Eve act independently.
- So, the probability of the 1st case is $p_1 = a_+ \cdot b_+ \cdot e_\times$, where:
  - a_+ is the probability that Alice selects $+$,
  - b_+ is the probability that Bob selects $+$,
  - $e_\times$ is the probability that Eve selects $\times$.

94. What Do We Want to Maximize (cont-d)

- Similarly, the probability p_2 of the 2nd case is $p_1 = a_\times \cdot b_\times \cdot e_+$
- These two cases are incompatible.
- So the overall probability p of detecting the eavesdropper is the sum of the above two probabilities:

$$p = a_+ \cdot b_+ \cdot e_\times + a_\times \cdot b_\times \cdot e_+.$$

- Taking into account that $a_\times = 1 - a_+$, $b_\times = 1 - b_+$, and $e_\times = 1 - e_+$, we get:

$$p = a_+ \cdot b_+ \cdot (1 - e_+) + (1 - a_+) \cdot (1 - b_+) \cdot e_+.$$

- This probability depends on Eve's selection e_+.
- We want to maximize the worst-case probability of detection, when Eve uses her best strategy:

$$J = \min_{e_+ \in [0,1]} \{a_+ \cdot b_+ \cdot (1 - e_+) + (1 - a_+) \cdot (1 - b_+) \cdot e_+\}.$$

95. Analyzing the Optimization Problem

- Once the values a_+ and b_+ are fixed, the expression that Eve wants to minimize is a linear function of e_+:

$$p = a_+ \cdot b_+ - a_+ \cdot b_+ \cdot e_+ + (1 - a_+) \cdot (1 - b_+) \cdot e_+ =$$

$$a_+ \cdot b_+ + e_+ \cdot ((1 - a_+) \cdot (1 - b_+) - a_+ \cdot b_+).$$

- We want to minimize this expression over all possible values of e_+ from the interval $[0, 1]$.
- A linear function on an interval always attains its min at one of the endpoints.
- Thus, to find the minimum of the above expression over e_+, it is sufficient:
  - to consider the two endpoints $e_+ = 0$ and $e_+ = 1$ of this interval, and
  - take the smallest of the resulting two values.

96. Analyzing the Optimization Problem (cont-d)

- For $e_+ = 0$, the expression becomes $a_+ \cdot b_+$.
- For $e_+ = 1$, the expression becomes $(1 - a_+) \cdot (1 - b_+)$.
- Thus, the minimum of the expression can be equivalently described as:

$$J = \min\{a_+ \cdot b_+, (1 - a_+) \cdot (1 - b_+)\}.$$

- We need to find the values a_+ and b_+ for which this quantity attains its largest possible value.
- Let us first, for each a_+, find the value b_+ for which the J attains its maximum possible value.
- In the formula for J, $a_+ \cdot b_+$, is increasing from 0 to a_+ as b_+ goes from 0 to 1.
- The second expression $(1-a_+)\cdot(1-b_+)$ decreases from $1 - a_+$ to 0 as b_+ goes from 0 to 1.

97. Analyzing the Optimization Problem (cont-d)

- Thus, for small b_+, the first of the two expressions is smaller.
- So, for these b_+, $J = a_+ \cdot b_+$ and is, thus, increasing with b_+;
- For larger b_+, the second of the two expressions is smaller.
- Thus for these b_+, $J = (1 - a_+) \cdot (1 - b_+)$ and is, so, decreasing with b_+.
- So J first increases and then decreases.
- Thus, its maximum is attained at a point when J switches from increasing to decreasing, i.e., where:

$$a_+ \cdot b_+ = (1 - b_+) \cdot (1 - a_+), \text{ i.e.,}$$

$$a_+ \cdot b_+ = 1 - a_+ - b_+ + a_+ \cdot b_+, \text{ so } b_+ = 1 - a_+.$$

98. Analyzing the Optimization Problem (cont-d)

- Substituting $b_+ = 1 - a_+$ into the formula for J, we get

$$J = \min\{a_+ \cdot (1 - a_+), (1 - a_+) \cdot a_+\} = a_+ \cdot (1 - a_+).$$

- We want to find the value a_+ that maximizes this expression: it is $a_+ = 0.5$.
- Since $b_+ = 1 - a_+$, we get $b_+ = 1 - 0.5 = 0.5$.
- Thus, the current quantum cryptography algorithm is indeed optimal.
- Similar arguments show:
  - that the best is to use 45 degrees rotation, and
  - that the best is to have 0s and 1s in b_i with probability 0.5.

99. Another Issue: Need for Parallel Quantum Computing

- While quantum computing is fast, its speeds are also limited.
- To further speed up computations, a natural idea is to have several quantum computers working in parallel.
- Then each of them solves a part of the problem.
- This idea is similar to how we humans solve complex problems:
  - if a task is too difficult for one person to solve – be it building a big house or proving a theorem,
  - several people team up and together solve the task.

100. Need for Teleportation

- To successfully collaborate, quantum computers need to exchange intermediate states of their computations.
- Here lies a problem: for complex problems, we would like to use computers in different geographic areas.
- However, a quantum state gets changed when it is sent far away.
- Researchers have come up with a way to avoid this sending, called *teleportation*.
- There exists a scheme for teleportation.

101. Problem

- It is not clear how good is the current teleportation scheme.
- Maybe there are other schemes which are faster (or better in some other sense)?
- In this tutorial, we show that the existing teleportation scheme is, in some reasonable sense, unique.
- In this sense, this sense is the best.
- To explain this result, we start by a brief reminder of the basics of quantum physics.

102. Basic States in Quantum Physics

- In quantum physics:
  - in addition to the usual (non-quantum) states $s_1, s_2, \ldots,$
  - we also have *superpositions* of these states, i.e., states of the type $\alpha_1 \cdot s_1 + \alpha_2 \cdot s_2 + \ldots$
- Here $\alpha_1, \alpha_2, \ldots$ are complex numbers (called *amplitudes*) for which $|\alpha_1|^2 + |\alpha_2|^2 + \ldots = 1.$
- For example, a computer is formed from devices representing *binary digits* (*bits*, for short).
- These devices can be in two possible states: 0 and 1.
- In quantum physics, we also have superpositions

$$\alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle, \text{ where } |\alpha_0|^2 + |\alpha_1|^2 = 1.$$

- The corresponding quantum system is known as a *quantum bit*, or *qubit*, for short.

103. Composite States in Quantum Physics

- There is a straightforward way to describe a composite system consisting of two independent subsystems.
- Due to independence, to describe the set of the system as a whole, it is sufficient to describe:
  - the state s of the first subsystem and
  - the state s' of the second subsystem.
- Thus, a state of the system as a whole is an ordered pair $\langle s, s' \rangle$ of the two states; let us denote:
  - possible states of the 1st subsystem by $s_1, s_2, \ldots$;
  - possible states of the 2nd subsystem by $s'_1, s'_2, \ldots$
- The subsystems are independent.
- So, the possible states of the 1st subsystem do not depend on the state of the 2nd.

104. Composite States (cont-d)

- Thus, the set of all states of the system as a whole is the set of all possible pairs $\langle s_i, s'_j\rangle$.
- The set of all such pairs is known as the *Cartesian product*; it is denoted by $\{s_1, s_2, \ldots\} \times \{s'_1, s'_2, \ldots\}$.
- These notations are usually simplified: e.g., $\langle 0, 1\rangle$ is denoted simply as 01.
- In quantum physics, we can also have superpositions of such states, i.e., the states of the type

$$\alpha_{11}\cdot\langle s_1, s'_1\rangle+\alpha_{12}\cdot\langle s_1, s'_2\rangle+\ldots+\alpha_{21}\cdot\langle s_2, s'_1\rangle+\alpha_{22}\cdot\langle s_2, s'_2\rangle+\ldots$$

- Here, $|\alpha_{11}|^2 + |\alpha_{12}|^2 + \ldots + |\alpha_{21}|^2 + |\alpha_{22}|^2 + \ldots = 1.$
- To describe such a state, we need to known all the values α_{ij}.
- These values form a matrix – i.e., in mathematical terms, a *tensor*.

105. Composite States (cont-d)

- Because of this fact, the set of all such states is known as the *tensor product* $S \otimes S'$, where:
  - S is the set of all possible quantum states of the first subsystem and
  - S' is the set of all possible quantum states of the second subsystem.
- So, the pair $\langle s, s' \rangle$ is denoted by $s \otimes s'$ and called a *tensor product* of the states s and s':
  - if the first subsystem is in the state s_i and the second subsystem is in the state s'_j,
  - then the state of the system is $\langle s_i, s'_j \rangle = s_i \otimes s'_j$.
- If $s = \alpha_1 \cdot s_1 + \alpha_2 \cdot s_2 + \ldots$ and $s' = \alpha'_1 \cdot s'_1 + \alpha'_2 \cdot s'_2 + \ldots$, then $s \oplus s' = \sum\limits_{i,j} \alpha_i \cdot \alpha'_j \cdot s_i \odot s'_j$.

106. Transformations in Quantum Physics

- Physically possible transformation are the mappings from state to state that satisfy the following properties:
  - superpositions get transformed into similar superpositions:

    $$T(\alpha_1 \cdot s_1 + \alpha_2 \cdots s_2 + \ldots) = \alpha_1 \cdot T(s_1) + \alpha_2 \cdot T(s_1) + \ldots,$$
  - $\sum |\alpha_i|^2 = 1$ is preserved: if $\sum |\alpha_i|^2 = 1$, then, for $T\left(\sum \alpha_i \cdot s_i\right) = \sum \beta_i \cdot s_i$, we have $\sum |\beta_i|^2 = 1$.
- Because of the first property, transformations are linear: $\sum \alpha_i \cdot s_i \to \sum \beta_i \cdot s_i$, with $\beta_i = \sum\limits_j t_{ij} \cdot \alpha_j$.
- Because of the second property, the matrix $T = (t_{ij})$ is *unitary*, i.e., $TT^\dagger = \mathbf{1}$, where $\mathbf{1}$ is a unit matrix.
- Here, $T^\dagger \stackrel{\text{def}}{=} (t^*_{ji})$, with z^* denoting the complex conjugate number $(a + b \cdot \mathrm{i})^* \stackrel{\text{def}}{=} a - b \cdot \mathrm{i}$.

107. Measurement Process in Quantum Physics

- For binary states $\alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle$, if we want to measure whether the state is 0 or 1, then:
  - with probability $|\alpha_0|^2$, we get the result 0 – and the state turns into $|0\rangle$; and
  - with probability $|\alpha_1|^2$, we get the result 1 – and the state turns into $|1\rangle$.
- Since the result is either 0 or 1, the probabilities should add up to 1.
- This explains why physically possible states should satisfy the condition $|\alpha_0|^2 + |\alpha_1|^2 = 1$.
- In general, in a quantum state $\sum \alpha_i \cdot s_i$, we get s_i with probability $|\alpha_i|^2$.
- Once the measurement process detects the state s_i, the actual state turns into s_i.

108. Measurement Process (cont-d)

- Instead of the classical states s_i, we can use any orthonormal sequence of states $s'_i = \sum_j t_{ij} \cdot s_j$:
  - for each i, we have $\|s'_i\|^2 = 1$, where $\|s'_i\|^2 \stackrel{\text{def}}{=} \sum_j |t_{ij}|^2$ (*normal*), and
  - for each i and i', we have $s'_i \perp s'_{i'}$, i.e., $\langle s'_i | s'_{i'} \rangle = 0$, where $\langle s'_i | s'_{i'} \rangle \stackrel{\text{def}}{=} \sum_j t_{ij} \cdot t^*_{i'j}$ (*orthogonal*).
- In a state $\sum \alpha'_i \cdot s'_i$, with probability $|\alpha'_i|^2$, the measurement result is s'_i and the state turns into s'_i.
- In general, instead of orthogonal vectors, we can have a sequence of orthogonal linear spaces $L_1, L_2, \ldots$
- Here $L_i \perp L_j$ means that $s_i \in L_i$ and $s_j \in L_j$ implies

$$s_i \perp s_j.$$

109. Measurement Process (cont-d)

- In this case, every state s can be represented as a sum $s = \sum s_i$ of the vectors $s_i \in L_i$.
- As a result of the measurement, with probability $\|s_i\|^2$:
  - we conclude that the state is in the space L_i, and
  - the original state turns into a new state $s_i / \|s_i\|$.

110. Need for Communication

- At one location, we have a particle in a certain state.
- We want to send this state to some other location.
- Usually, the sender is denoted by A and the receiver by B.
- In communications, it is common to call the sender Alice, and to call the receiver Bob:
  - states corresponding to Alice are usually described by using a subscript A, and
  - states corresponding to Bob are usually described by using a subscript B.

111. Communication Is Straightforward in Classical Physics

- In classical (pre-quantum) physics, the communication problem has a straightforward solution.
- If we want to communicate a state:
  - we measure all possible characteristics of this state,
  - send these values to Bob, and
  - let Bob reproduce the object with these characteristics.
- This is how, e.g., 3D printing works.
- This solution is based on the fact that:
  - in classical (non-quantum) physics
  - we can, in principle, measure all characteristic of a system without changing it.

112. Communication Is a Challenge in Quantum Physics

- The problem is that in quantum physics, such a straightforward approach is not possible.
- In quantum physics, every measurement changes the state.
- Moreover, each measurement irreversibly deletes some information about the state.
- For example, if we start with a state $\alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle$, all we get after the measurement is either 0 or 1.
- There is no way to reconstruct the values α_0 and α_1 that characterize the original state.
- Since we cannot use a direct approach for communicating a state, we need to use an indirect approach.
- This approach is known as *teleportation*.

113. What We Consider in This Tutorial

- We consider the quantum analogue of the simplest possible non-quantum state.
- The simplest case when communication is needed is when the system can be in two different states.
- In the computer, such situation can be naturally described if we associate these states with 0 and 1.
- Alice has a state $\alpha_0 \cdot |0\rangle + \alpha_1 \cdot |1\rangle$ that she wants to communicate to Bob.
- The above state is not exclusively Alice's or Bob's.
- So, to describe this state, we will use the next letter C.
- In these terms, Alice has a state $\alpha_0 \cdot |0\rangle_C + \alpha_1 \cdot |1\rangle_C$.
- She wants to communicate this state to Bob.

114. Preparing for Teleportation: an Entangled State

- To make teleportation possible, Alice and Bob prepare a special *entangled* state:
$$\frac{1}{\sqrt{2}} \cdot |0_A 1_B\rangle + \frac{1}{\sqrt{2}} \cdot |1_A 0_B\rangle.$$
- This state is a superposition of two classical states:
  - the state $0_A 1_B$ in which A is in state 0 and B is in state 1, and
  - the state $1_A 0_B$ in which A is in state 1 and B is in state 0.
- At first, the state C is independent of A and B.
- So, the joint state is a tensor product of the AB-state and the C-state:
$$\frac{\alpha_0}{\sqrt{2}} \cdot |0_A 1_B 0_C\rangle + \frac{\alpha_1}{\sqrt{2}} \cdot |0_A 1_B 1_C\rangle + \frac{\alpha_0}{\sqrt{2}} \cdot |1_A 0_B 0_C\rangle + \frac{\alpha_1}{\sqrt{2}} \cdot |1_A 0_B 1_C\rangle.$$

115. First Stage: Measurement

- First, Alice performs a measurement procedure on the parts A and C which are available to her.
- We perform the measurement w.r.t. $L_i = L_B \otimes t_i$.
- Here, L_B is the set of all possible linear combinations of $|0\rangle_B$ and $|1\rangle_B$.
- The states t_i are as follows:

$$t_1 = \frac{1}{\sqrt{2}} \cdot |0_A 0_C\rangle + \frac{1}{\sqrt{2}} \cdot |1_A 1_C\rangle;$$

$$t_2 = \frac{1}{\sqrt{2}} \cdot |0_A 0_C\rangle - \frac{1}{\sqrt{2}} \cdot |1_A 1_C\rangle;$$

$$t_3 = \frac{1}{\sqrt{2}} \cdot |0_A 1_C\rangle + \frac{1}{\sqrt{2}} \cdot |1_A 0_C\rangle;$$

$$t_4 = \frac{1}{\sqrt{2}} \cdot |0_A 1_C\rangle - \frac{1}{\sqrt{2}} \cdot |1_A 0_C\rangle.$$

116. First Stage: Measurement (cont-d)

- One can easily check that the states t_i are orthonormal, hence the spaces L_i are orthogonal.
- Let's represent the state in as $s = \sum s_i$, with $s_i \in L_i$:

$$s_1 = \left(\frac{\alpha_0}{2} \cdot |1_B\rangle + \frac{\alpha_1}{2} |0_B\rangle\right) \otimes t_1,$$

$$s_2 = \left(\frac{\alpha_0}{2} \cdot |1_B\rangle - \frac{\alpha_1}{2} \cdot |0_B\rangle\right) \otimes t_2,$$

$$s_3 = \left(\frac{\alpha_1}{2} \cdot |1_B\rangle + \frac{\alpha_0}{2} \cdot |0_B\rangle\right) \otimes t_3,$$

$$s_4 = \left(\frac{\alpha_1}{2} \cdot |1_B\rangle - \frac{\alpha_0}{2} \cdot |0_B\rangle\right) \otimes t_4.$$

- Here, for each i, we have $\|s_i\| = \dfrac{1}{2}$.

117. First Stage: Measurement (cont-d)

- So, with equal probability of $\dfrac{1}{4}$, we get one of the following four states – and Alice knows which one it is:

$$(\alpha_0 \cdot |1_B\rangle + \alpha_1 \cdot |0_B\rangle) \otimes t_1;$$

$$(\alpha_0 \cdot |1_B\rangle - \alpha_1 \cdot |0_B\rangle) \otimes t_2;$$

$$(\alpha_1 \cdot |1_B\rangle + \alpha_0 \cdot |0_B\rangle) \otimes t_3;$$

$$(\alpha_1 \cdot |1_B\rangle - \alpha_0 \cdot |0_B\rangle) \otimes t_4.$$

118. Two Final Stages

- Alice sends to Bob the measurement result.
- So, Bob knows in which the four states the system is.
- Bob performs a transformation of his state B.
- In the first case, he uses a unitary transformation that swaps $|0\rangle_B$ and $|1\rangle_B$: $t_{01} = t_{10} = 1$ and $t_{00} = t_{11} = 0$.
- In the second case, he uses a unitary transformation for which $t_{01} = 1$, $t_{10} = -1$ and $t_{00} = t_{11} = 0$.
- In the third case, he already has the desired state.
- In the fourth case, he uses a unitary transformation for which $t_{00} = -1$, $t_{11} = 1$, and $t_{01} = t_{10} = 0$.
- As a result, in all fours cases, he gets the original state $\alpha_0 \cdot |0\rangle_B + \alpha_1 \cdot |1\rangle_B$.

119. Formulation of the Problem

- Teleportation is possible because we have prepared an *entangled* state.
- This is a state s_{AB} in which the states of Alice and Bob are not independent.
- However, the above is not the only possible entangled state.
- Let us consider, instead, a general joint state of two qubits:

$$a_{00} \cdot |0_A 0_B\rangle + a_{01} \cdot |0_A 1_B\rangle + a_{10} \cdot |1_A 0_B\rangle + a_{11} \cdot |1_A 1_B\rangle.$$

- What will happen if we use this more general entangled state?

120. Analysis of the Problem

- For the general state, the joint state of all three subsystems has the form

$$\alpha_0 \cdot a_{00} \cdot |0_A 0_B 0_C\rangle + \alpha_1 \cdot a_{00} \cdot |0_A 0_B 1_C\rangle +$$
$$\alpha_0 \cdot a_{01} \cdot |0_A 1_B 0_C\rangle + \alpha_1 \cdot a_{01} \cdot |0_A 1_B 1_C\rangle +$$
$$\alpha_0 \cdot a_{10} \cdot |1_A 0_B 0_C\rangle + \alpha_1 \cdot a_{10} \cdot |1_A 0_B 1_C\rangle +$$
$$\alpha_0 \cdot a_{11} \cdot |1_A 1_B 0_C\rangle + \alpha_1 \cdot a_{11} \cdot |1_A 1_B 1_C\rangle.$$

- Substituting expressions for s_i, we get $s = S_1 \otimes t_1 + S_2 \otimes t_2 + \ldots$, where:

$$S_1 = \left(\frac{\alpha_0 \cdot a_{00}}{\sqrt{2}} + \frac{\alpha_1 \cdot a_{10}}{\sqrt{2}}\right) \cdot |0\rangle_B + \left(\frac{\alpha_0 \cdot a_{01}}{\sqrt{2}} + \frac{\alpha_1 \cdot a_{11}}{\sqrt{2}}\right) \cdot |1\rangle_B.$$

- $S_2, \ldots$ are described by similar expressions.
- This means that after the measurement, Bob will have the normalized state $S_1/\|S_1\|$.

121. Analysis of the Problem (cont-d)

- To perform teleportation, we need to transform this state into the original state $\alpha_0 \cdot |0\rangle_B + \alpha_1 \cdot |1\rangle_B$.
- Thus, the transformation from the resulting state $S_1/\|S_1\|$ to the original state must be unitary.
- It is known that the inverse transformation to a unitary one is also unitary.
- In general, a unitary transformation transforms orthonormal states into orthonormal ones.

122. Analysis of the Problem (cont-d)

- So, the inverse transformation:
  - maps the state $|0\rangle_B$ (corresponding to $\alpha_0 = 1$ and $\alpha_1 = 0$) into a new state
  $$|1'\rangle_B \stackrel{\text{def}}{=} \text{const} \cdot (a_{00} \cdot |0\rangle_B + a_{01} \cdot |1\rangle_B),$$
  - maps the state $|1\rangle_B$ (corresponding to $\alpha_0 = 0$ and $\alpha_1 = 1$) into a new state
  $$|0'\rangle_B \stackrel{\text{def}}{=} \text{const} \cdot (a_{10} \cdot |0\rangle_B + a_{11} \cdot |1\rangle_B).$$
- It should transform two original orthonormal vectors $|0\rangle_B$, $|1\rangle_B$ into two new orthonormal ones $|0'\rangle_B$, $|1'\rangle_B$.
- In terms of these new states, the entangled state is
$$\text{const} \cdot (|0\rangle_A \otimes |1'\rangle_B + |1\rangle_B \otimes |0'\rangle_B).$$
- The sum of the squares of absolute values of all the coefficients should add up to 1.

123. Analysis of the Problem (cont-d)

- Then const $= \dfrac{1}{\sqrt{2}}$, and the entangled state takes the familiar form $\dfrac{1}{\sqrt{2}} \cdot (|0\rangle_A \otimes |1'\rangle_B + |1\rangle_B \otimes |0'\rangle_B)$.
- This is exactly the entangled state used in the standard teleportation algorithm.

124. Quantum Part: Conclusion

- From the technical viewpoint:
  - the only entangled state that leads to a successful teleportation
  - is the state corresponding to the standard quantum teleportation algorithm,
  - for some orthornomal states $|0'\rangle_B$ and $|1'\rangle_B$.
- Thus, we have shown that, indeed, the existing quantum teleportation algorithm is unique.
- So we should not waste our time and effort looking for more efficient alternative teleportation algorithms.

Main Objective
Time to Gather Stones
Case Studies
Fuzzy Case
Neural Network Case
Quantum Computing
Proofs (if time allows)
Home Page
Title Page
◀◀ ▶▶
◀ ▶
Page 128 of 147
Go Back
Full Screen
Close
Quit

Part IV

Proofs (if time allows)

125. Why Fractional Linear

- Every transformation is a composition of infinitesimal ones $x \to x + \varepsilon \cdot f(x)$, for infinitely small ε.
- So, it's enough to consider infinitesimal transformations.
- The class of the corresponding functions $f(x)$ is known as a *Lie algebra* A of the corresponding transformation group.
- Infinitesimal linear transformations correspond to $f(x) = a + b \cdot x$, so all linear functions are in A.
- In particular, $1 \in A$ and $x \in A$.
- For any λ, the product $\varepsilon \cdot \lambda$ is also infinitesimal, so we get $x \to x + (\varepsilon \cdot \lambda) \cdot f(x) = x \to x + \varepsilon \cdot (\lambda \cdot f(x))$.
- So, if $f(x) \in A$, then $\lambda \cdot f(x) \in A$.

126. Why Fractional Linear (cont-d)

- If we first apply $f(x)$, then $g(x)$, we get

  $x \to (x+\varepsilon\cdot f(x))+\varepsilon\cdot g(x+\varepsilon\cdot f(x)) = x+\varepsilon\cdot(f(x)+g(x))+o(\varepsilon).$

- Thus, if $f(x) \in A$ and $g(x) \in A$, then $f(x)+g(x) \in A$.
- So, A is a linear space.
- In general, for the composition, we get

$$x \to (x+\varepsilon_1\cdot f(x))+\varepsilon_2\cdot g(x_1+\varepsilon_1\cdot f(x)) =$$

  $x+\varepsilon_1\cdot f(x)+\varepsilon_2\cdot g(x)+\varepsilon_1\cdot\varepsilon_2\cdot g'(x)\cdot f(x)+$ quadratic terms.

- If we then apply the inverses to $x \to x+\varepsilon_1\cdot f(x)$ and $x \to x+\varepsilon_2\cdot g(x)$, the linear terms disappear, we get:

$$x \to x+\varepsilon_1\cdot\varepsilon_2\cdot\{f,g\}(x),\ \text{where } \{f,g\} \stackrel{\text{def}}{=} f'(x)\cdot g(x)-f(x)\cdot g'(x).$$

- Thus, if $f(x) \in A$ and $g(x) \in A$, then $\{f,g\}(x) \in A$.
- The expression $\{f,g\}$ is known as the*Poisson bracket*.

127. Why Fractional Linear (cont-d)

- Let's expand any function $f(x)$ in Taylor series:

$$f(x) = a_0 + a_1 \cdot x + \ldots$$

- If k is the first non-zero term in this expansion, we get

$$f(x) = a_k \cdot x^k + a_{k+1} \cdot x^{k+1} + a_{k+2} \cdot x^{k+2} + \ldots$$

- For every λ, the algebra A also contains

$$\lambda^{-k} \cdot f(\lambda \cdot x) = a_k \cdot x^k + \lambda \cdot a_{k+1} \cdot x^{k+1} + \lambda^2 \cdot a_{k+2} \cdot x^{k+2} + \ldots$$

- In the limit $\lambda \to 0$, we get $a_k \cdot x^k \in A$, hence $x^k \in A$.
- Thus, $f(x) - a_k \cdot x^k = a_{k+1} \cdot x^{k+1} + \ldots \in A$.
- We can similarly conclude that A contains all the terms x^n for which $a_n \neq 0$ in the original Taylor expansion.

128. Why Fractional Linear (cont-d)

- Since $g(x) = 1 \in A$, for each $f \in A$, we have
$$\{f, 1\} = f'(x) \cdot 1 + f(x) \cdot q' = f'(x) \in A.$$
- Thus, for each k, if $x^k \in A$, we have $(x^k)' = k \cdot x^{k-1} \in A$ hence $x^{k-1} \in A$, etc.
- Thus, if $x^k \in A$, all smaller power are in A too.
- In particular, this means that if $x^k \in A$ for some $k \geq 3$, then we have $x^3 \in A$ and $x^2 \in A$; thus:
$$\{x^3, x^2\} = (x^3)' \cdot x^2 - x^3 \cdot (x^2)' = 3 \cdot x^2 \cdot x^2 - x^3 \cdot 2 \cdot x = x^4 \in A.$$
- In general, once $x^k \in A$ for $k \geq 3$, we get
$$\{x^k, x^2\} = (x^k)' \cdot x^2 - x^k \cdot (x^2)' = k \cdot x^{k-1} \cdot x^2 - x^k \cdot 2 \cdot x =$$
$$(k-2) \cdot x^{k+1} \in A \text{ hence } x^{k+1} \in A.$$
- So, by induction, $x^k \in A$ for all k.

129. Why Fractional Linear (cont-d)

- If $x^k \in A$ for some $k \geq 3$, then $x^k \in A$ for all k.
- Thus, A is infinite-dimensional – which contradicts to our assumption that A is finite-dimensional.
- So, we cannot have Taylor terms of power $k \geq 3$; therefore we have:

$$x \to x + \varepsilon \cdot (a_0 + a_1 \cdot x + a_2 \cdot x^2).$$

- This corresponds to an infinitesimal fractional-linear transformation

$$x \to \frac{\varepsilon \cdot A + (1 + \varepsilon \cdot B) \cdot x}{1 + \varepsilon \cdot D \cdot x} =$$
$$(\varepsilon \cdot A + (1 + \varepsilon \cdot B) \cdot x) \cdot (1 - \varepsilon \cdot D \cdot x) + o(\varepsilon) =$$
$$x + \varepsilon \cdot (A + (B - D) \cdot x - D \cdot x^2).$$

- So, to match, we need

$$A = a_0, \quad D = -a_2, \text{ and } B = a_1 - a_2.$$

130. Why Fractional Linear: Final Part

- We concluded that every infinitesimal transformation is fractionally linear.
- Every transformation is a composition of infinitesimal ones.
- Composition of fractional-linear transformations is fractional linear.
- Thus, all transformations are fractional linear.

131. Pooling: General Part of the Two Proofs

- Let us first prove that the optimal operation $*_{\rm opt}$ is itself scale-invariant: $R_\lambda(*_{\rm opt}) = *_{\rm opt}$ for all $\lambda > 0$.
- The fact that $*_{\rm opt}$ is optimal means that $* \preceq *_{\rm opt}$ for all $*$.
- In particular, $R_{\lambda^{-1}}(*) \preceq *_{\rm opt}$ for all $*$.
- Due to scale-invariance of the optimality criterion, this implies that $* \preceq R_\lambda(*_{\rm opt})$ for all $*$.
- Thus, the operation $R_\lambda(*_{\rm opt})$ is also optimal.
- But since the optimality criterion is final, there is only one optimal operation, so $R_\lambda(*_{\rm opt}) = *_{\rm opt}$.
- Scale-invariance is proven.
- Shift-invariance is proven similarly.
- For Proposition 2, we can similarly prove that the optimal $*$ is weakly shift-invariant: $W_{a_0}(*_{\rm opt}) = *_{\rm opt}$.

132. Proof of Proposition 1

- Let $a * b$ be the optimal combination operation.
- We have shown that this operation is scale-invariant and shift-invariant.
- Let us prove that it has one of the above two forms.
- For every pair (a, b), we can have three different cases: $a = b$, $a < b$, and $a > b$.
- Let us consider them one by one.
- Let us first consider the case when $a = b$.
- Let us denote $v \stackrel{\text{def}}{=} 1 * 1$.
- From scale-invariance with $\lambda = 2$, from $1 * 1 = v$, we get $2 * 2 = 2v$.
- From shift-invariance with $s = 1$, from $1 * 1 = v$, we get $2 * 2 = v + 1$.

133. Proof of Proposition 1 (cont-d)

- Thus, $2v = v + 1$, hence $v = 1$, and $1 * 1 = 1$.
- For $a > 0$, by applying scale-invariance with $\lambda = a$ to the formula $1 * 1 = 1$, we get $a * a = a$.
- For $a = 0$, if we denote $c \stackrel{\text{def}}{=} 0 * 0$, then, by applying shift-invariance with $s = 1$ to $0 * 0 = c$, we get

$$1 * 1 = c + 1.$$

- Since we already know that $1 * 1 = 1$, this means that $c + 1 = 1$ and thus, that $c = 0$, i.e., that $0 * 0 = 0$.
- So, for all $a \geq 0$, we have $a * a = a$.
- In this case, $\min(a, a) = \max(a, a) = a$, so we have $a * a = \min(a, a)$ and $a * a = \max(a, a)$.
- Let us now consider the case when $a < b$. In this case, $b - a > 0$.

134. Proof of Proposition 1 (cont-d)

- Let us denote $t \stackrel{\text{def}}{=} 0 * 1$.
- By applying scale-invariance with $\lambda = b - a > 0$ to the formula $0 * 1 = t$, we get $0 * (b - a) = (b - a) \cdot t$.
- Now, by applying shift-invariance with $s = a$ to this formula, we get $a * b = (b - a) \cdot t + a$.
- To find possible values of t, let us take into account that the combination operation should be associative.
- This means, in particular, that for all possible triples a, b, and c for which we have $a < b < c$, we must have

$$a * (b * c) = (a * b) * c.$$

- Since $b < c$, by the above formula, we have $b * c = (c - b) * t + b$.
- Since $t \geq 0$, we have $b * c \geq b$ and thus, $a < b * c$.

135. Proof of Proposition 1 (cont-d)

- So, to compute $a * (b * c)$, we can also use the above formula, and get $a * (b * c) = (b * c - a) \cdot t + a =$
$$((c - b) \cdot t + b) \cdot t + a = c \cdot t^2 + b \cdot (t - t^2) + a.$$
- Let us restrict ourselves to the case when $a * b < c$.
- In this case, the general formula implies that
$$(a*b)*c = (c-a*b)\cdot t + a*b = (c-((b-a)\cdot t + a))\cdot t + (b-a)\cdot t + a.$$
- So $(a * b) * c = c \cdot t + b \cdot (t - t^2) + a \cdot (1 - t)^2$.
- Due to associativity, the two formulas must coincide for all a, b, and c for which $a < b < c$ and $c > a * b$.
- These two linear expressions must be equal for all sufficiently large values of c.
- Thus, the coefficients at c must be equal, i.e., we must have $t = t^2$.

136. Proof of Proposition 1 (cont-d)

- From $t = t^2$, we conclude that $t - t^2 = t \cdot (1 - t) = 0$, so either $t = 0$ or $1 - t = 0$ (in which case $t = 1$).
- If $t = 0$, then the above formula has the form $a * b = a$, i.e., since $a < b$, the form $a * b = \min(a, b)$.
- If $t = 1$, then the above formula has the form

$$a * b = (b - a) + a = b.$$

- Since $a < b$, we get $a * b = \max(a, b)$.
- If $a > b$, then, by commutativity, we have $a * b = b * a$, where now $b < a$.
- So, either we have $a * b = \min(a, b)$ for all a and b, or we have $a * b = \max(a, b)$ for all a and b.
- The proposition is proven.

137. Proof of Proposition 2

- Let $a * b$ be the optimal combination operation.
- We have proven that this operation is scale-invariant and weakly shift-invariant.
- This means that $a * b = c$ implies $(a + s) * (b + s) = c + f(s)$.
- Let us prove that the optimal operation $*$ has one of the above four forms.
- Let us first prove that $0 * 0 = 0$.
- Indeed, let s denote $0 * 0$.
- Due to scale-invariance, $0 * 0 = s$ implies that $(2 \cdot 0) * (2 \cdot 0) = 2s$, i.e., that $0 * 0 = 2s$.
- So, we have $s = 2s$, hence $s = 0$ and $0 * 0 = 0$.
- Similarly, if we denote $v \stackrel{\text{def}}{=} 1 * 1$, then, due to scale-invariance with $\lambda = a$, $1 * 1 = v$ implies that $a * a = v \cdot a$.

138. Proof of Proposition 2 (cont-d)

- On the other hand, due to weak shift-invariance with $a_0 = a$, $0 * 0 = 0$ implies that $a * a = f(a)$.
- Thus, we conclude that $f(a) = v \cdot a$.
- Let us now consider the case when $a < b$ and, thus, $b - a > 0$.
- Let us denote $t \stackrel{\text{def}}{=} 0 * 1$.
- From scale-invariance with $\lambda = b - a$, from $0 * 1 = t \geq 0$, we get $0 * (b - a) = t \cdot (b - a)$.
- From weak shift-invariance with $a_0 = a$, we get $a * b = t \cdot (b - a) + v \cdot a$, i.e., $a * b = t \cdot b + (v - t) \cdot a$.
- The combination operation should be associative: $a * (b * c) = (a * b) * c$.
- When $b < c$, we have $b * c = t \cdot c + (v - t) \cdot b$.

139. Proof of Proposition 2 (cont-d)

- We know that $t \geq 0$. This means that we have either $t > 0$ and $t = 0$.
- Let us first consider the case when $t > 0$.
- In this case, for sufficiently large c, we have $b * c > a$.
- So, by applying the above formula to a and $b * c$, we conclude that

  $$a*(b*c) = t\cdot(b*c)+(v-t)\cdot a = t^2\cdot c+t\cdot(v-t)\cdot b+(v-t)\cdot a.$$

- For sufficient large c, we also have $a * b < c$.
- In this case, the general formula implies that

  $$(a*b)*c = (t\cdot b+(v-t)\cdot a)*c = t\cdot c+t\cdot(v-t)\cdot b+(v-t)^2\cdot a.$$

- Due to associativity, these formulas must coincide for all a, b, and c for which

  $$a < b < c, \quad c > a * b, \text{ and } b * c > a.$$

140. Proof of Proposition 2 (cont-d)

- These two linear expressions must be equal for all sufficiently large values of c.
- So, the coefficients at c must be equal, i.e., we must have $t = t^2$.
- From $t = t^2$, we conclude that $t - t^2 = t \cdot (1 - t) = 0$.
- Since we assumed that $t > 0$, we must have $t - 1 = 0$, i.e., $t = 1$.
- The coefficients at a must also coincide, so we must have $v - t = (v - t)^2$, hence either $v - t = 0$ or $v - t = 1$.
- In the first case, the above formula becomes $a * b = b$, i.e., $a * b = \max(a, b)$ for all $a \le b$.
- Since the operation $*$ is commutative, this equality is also true for $b \le a$ and is, thus, true for all a and b.

141. Proof of Proposition 2 (cont-d)

- In the second case, the above formula becomes $a * b = a + b$ for all $a \le b$.
- Due to commutativity, this formula holds for all a, b.
- Let us now consider the case when $t = 0$.
- In this case, the above formula takes the form $a * b = (v - t) \cdot a$.
- Here, $a * b \ge 0$, thus $v - t \ge 0$.
- If $v - t = 0$, this implies that $a * b = 0$ for all $a \le b$ and thus, due to commutativity, for all a and b.
- Let us now consider the remaining case when $v - t > 0$.
- In this case, if $a < b < c$, then for sufficiently large c, we have $a * b < c$, hence

  $$(a*b)*c = (v-t)\cdot(a*b) = (v-t)\cdot((v-t)\cdot a) = (v-t)^2\cdot a.$$

142. Proof of Proposition 2 (cont-d)

- On the other hand, here $b * c = (v - t) \cdot b$.
- So, for sufficiently large b, we have $(v - t) \cdot b > a$, thus

$$a * (b * c) = (v - t) \cdot a.$$

- Due to associativity, we have $(v - t)^2 \cdot a = (v - t) \cdot a$, hence $(v - t)^2 = v - t$.
- Since $v - t > 0$, we have $v - t = 1$.
- In this case, the above formula takes the form $a * b = a = \min(a, b)$ for all $a \le b$.
- Thus, due to commutativity, we have $a * b = \min(a, b)$ for all a and b.
- We have thus shown that the combination operation indeed has one of the four forms.
- Proposition 2 is therefore proven.