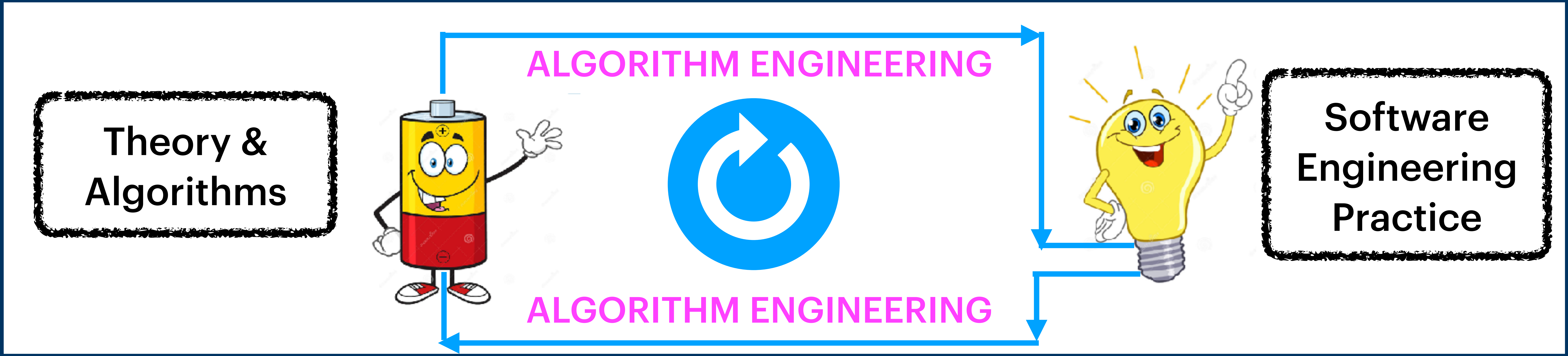


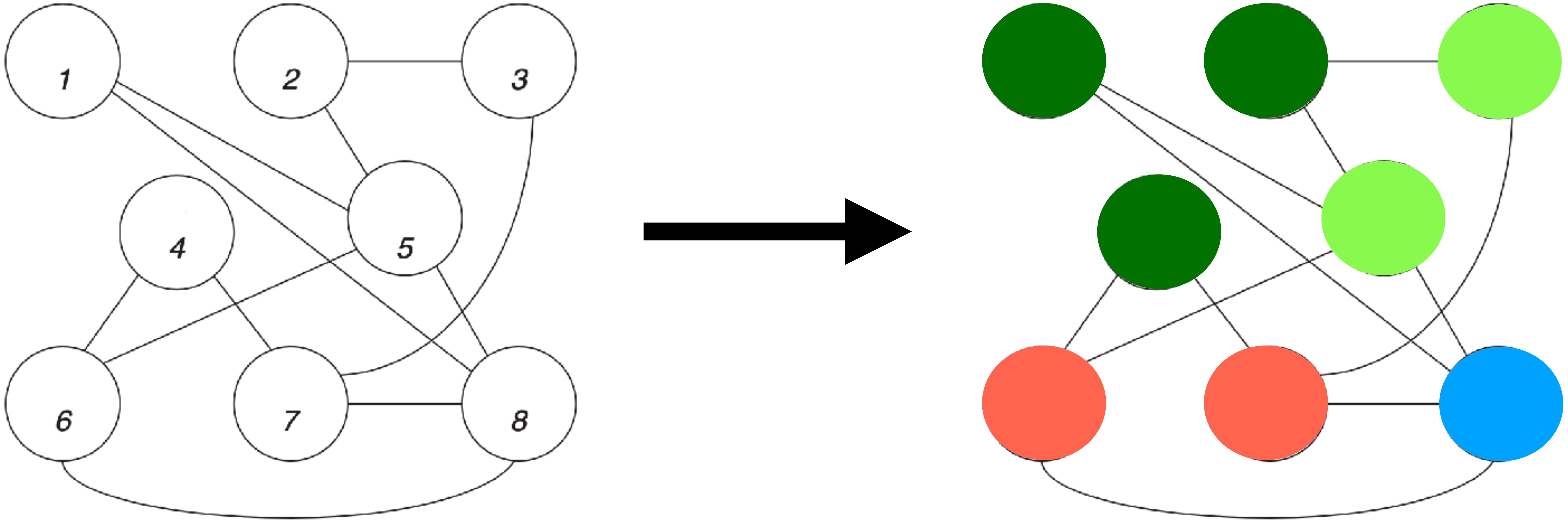
ALGORITHM ENGINEERING

Lecture 2:
Design of Experiments - I

M. Oğuzhan Külekci - kulekci@itu.edu.tr

The Need for Experiments

Graph coloring as an example case



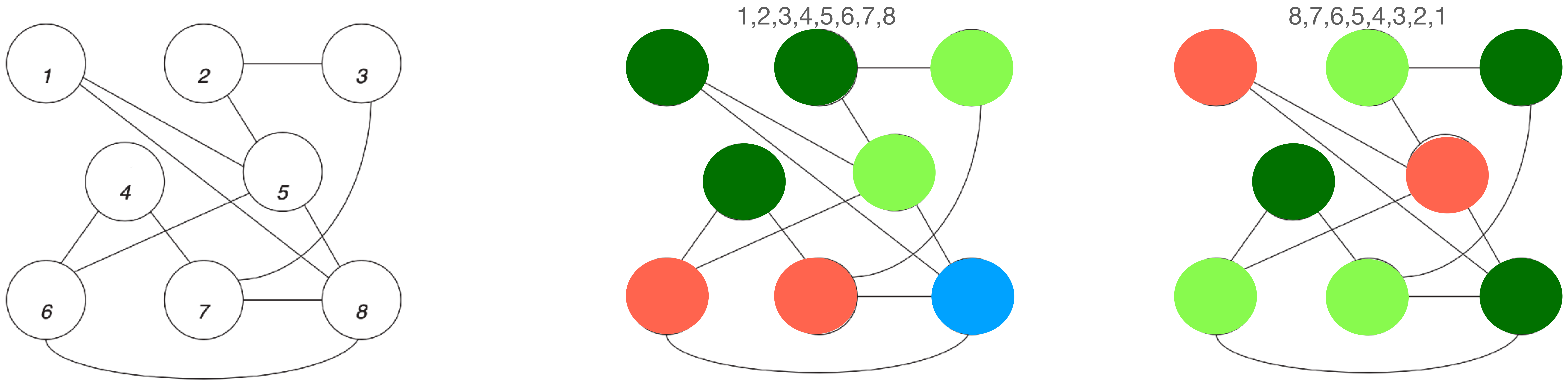
- Given a graph, assigning colors to vertices s.t. no adjacent vertices has the same color, in an NP-hard problem !
- Many practical scenarios, e.g. radio tower frequency assignment …

The Need for Experiments

Graph coloring as an example case

```
Greedy (G)
   for (v=1; v<=n; v++)
       for (c=1; c<=n; c++)
           if (G.checkColor(c, v)) {
               G.assignColor(c, v)
               break      // skip to next vertex
           }
   return G.colorCount()
```

- Different traversal of the vertices produce different results !
- How to find the minimum ?



The Need for Experiments

Graph coloring as an example case

```
Random (G, I)
    bestCount = Infinity
    bestColoring = null
    for (i=1; i<=I; i++){
         G.unColor()           //remove colors
         G.randomVertexOrder()
         count = Greedy(G)
         if (count < bestCount) {
             bestCount = count
             bestColoring = G.saveColoring()
         }
    }
    report (bestColor, bestCount)
```

- Brute-force needs n! trials.

- Execution time according to number of vertices and edges ?
- How many trials would provide good performace ?
- How does it compare with the best known algorithms?
- Any specific graphs favoring the greedy solution?
- What should be done to make inner operations speedy?
- ....

Experimental Design

A plan to answer a specific question

What we aim in experimentation ?

- Reproducibility: Others should be able to repeat it
- Efficiency : Results with minimum number of experiments
- Generality : Produce results to cover largest available
- Newsworthyness: Surprising results (not always …)
- Correctness : Right production of data
- Validity : Right conclusions

Experimental Design

Pilot study versus Workhorse

The scale of experimentation:

A) **Pilot study :** Basic exploration to understand what should be measured
Validity of assumptions, merit of the ideas, identify important points, eliminate details, technical details of the full experimentation (, e.g., required experimentation size, how long will it take, how the parameters should be set)

B) **Workhorse :** Complete experimentation to answer the questions

Leverage the pilot study to create better workhorse experiments.

Correctness and Validity

Spurious result: Arrive a wrong explanation due to some experimentation errors

File Name	n	m	GPR	CL	LC
R125.1.col	125	209	5	5	5
R125.5.col	125	7501	46	46	46
mulsol.i.1.col	197	3925	49	49	49
DSJ125.5.col	125	7782	20	18	17
DSJ250.5.col	250	31336	35	32	29
DSJ500.5.col	500	125248	65	57	52

Floor/ceiling effect: Easiest or hardest cases are questionable

Tip: Experiment with wider input space to observe differences !

Experimental artifacts: Due to some inherent calculation errors of the implementation

- Time measurement errors
- Floating number precision
- Bugs in the software,
- Random number generators

Tip: Replication - repeat the experiments on different platforms

Efficiency and Generality of Experimentation

Generality: We are testing against a scope. Does the experiments span this target space ?

Efficiency: Maximize the information gain per unit effort of experimentation

- Pilot phase explores efficient ways of workhorse experimentation phase
- Efficiency depends on the speed of the software, test environment, quality of the data, and generality of the conclusions.

Experimental Design Basics

Definitions and Concepts

Performance metric: Time, solution quality, space…

Performance indicator: The measured quantity, seconds, megabytes, returned result…

Parameter: Any property that effects the performance metric

- Algorithm parameter : Parameters of the tested solution, e.g. I in the graph coloring
- Instance parameter : The properties of the input data, number of edges, vertices, etc…
- Environment parameter : The operating system, compiler, etc…
- Noise parameter: Semi-controlled parameter, e.g. bit vector with a given polarity
- Fixed parameter: Never changing throughout the experiments.

Factor and Level: What will be manipulated in the experiment and in what levels

- e.g.,$data \in \{random, real\}, n \in \{100{,}200{,}300\}, I \in \{10^2{,}10^3{,}10^4\}$, etc…

Design points: Any combination of the levels of factors, e.g. $\{data = random, n = 300,\ I = 10^3\}$

Experimental Design Basics

Selecting the input class

Real versus artificially generated data

- **Stress test:** Boundary checking, e.g., empty graph, fully connected graph
- **Worst-case scenarios:** The input which is not in favor the solution, hardest case, quick sort on a sorted sequence
- **Random inputs:** Repeat the data generation with the same parameter several times
    - Notice that random inputs can generate data that is hard to find in real data sets
- **Publicly available testbed or test data:** Commonly assumed test data instances

Experimental Design Basics

Deciding on factors and design points

Why do we perform experiments ?

- **Assessment:** What is the general performance, bottlenecks, what parameters effect
- **Horse-race:** Which one is better ?
- **Parameter Fitting:** If the cost is $f(x) = an^2 + bn + c$, then find a, b, and c
- **Model fitting:** Find what is the best cost function, is it $f(x) = an + b$ or $f(x) = b \log n + c$

The pilot study reveals the most important parameters, which become the factors of the experiments.

The parameters that does not effect the performance should be fixed.

Observations during the pilot will provide some sense on the levels as well.

Experimental Design Basics

Assessment and comparison experiments

Decide on the performance metrics and their indicators.

Determine the most important factors effecting the performance.

Doubling experiments:

How much does the indicator change when you double the factor ?
Notice that usually this factor is the size n of the input.
n= N, 2N, 4N, 8N, ….

- If indicator does not change then the effect is constant. No need to investigate.
- If increment by a constant, then the relation is logarithmic, $O(\log n)$.
- If the indicator doubles as well, then the relation is linear, $O(n)$.
- $O(n \log n)$?
- $O(n^2)$?

Experimental Design Basics

Doubling experiment is good, but too general to describe **the functional relation between the parameters and the algorithm performance**.

Factors to analyse Random() procedure:
Option, n, m, I (?)

Assume max color needed is k.

Called n.k times

Called n times

```
Greedy (G)
   for (v=1; v<=n; v++)
      for (c=1; c<=n; c++)
         if (G.checkColor(c, v)) {
            G.assignColor(c, v)
            break      // skip to next vertex
         }
   return G.colorCount()
```

Grid approach: Brute-force assignment of levels
or
Scaled approach: Assign wisely,
e.g. $m = n(n-1)/2^{1+a}$, a= 1,2,3

```
Random (G, I)
   bestCount = Infinity
   bestColoring = null
   for (i=1; i<=I; i++){
      G.unColor()         //remove colors
      G.randomVertexOrder()
      count = Greedy(G)
      if (count < bestCount) {
         bestCount = count
         bestColoring = G.saveColoring()
      }
   }
   report (bestColor, bestCount)
```

m edges, n vertices

	checkColor	assignColor	Vertex color representation
Option a	O(mk + n)		Single color attribute
Option b	O(nk + m)		Color + forbiddenColor

Experimental Design Basics

Full-factorial design

- Run experiments on all design points, and explain the results.
- Perticularly useful in horse race experimentation

Factors	Experiments							
	1	2	3	4	5	6	7	8
F_1	-	+	-	+	-	+	-	+
F_2	-	-	+	+	-	-	+	+
F_3	-	-	-	-	+	+	+	+

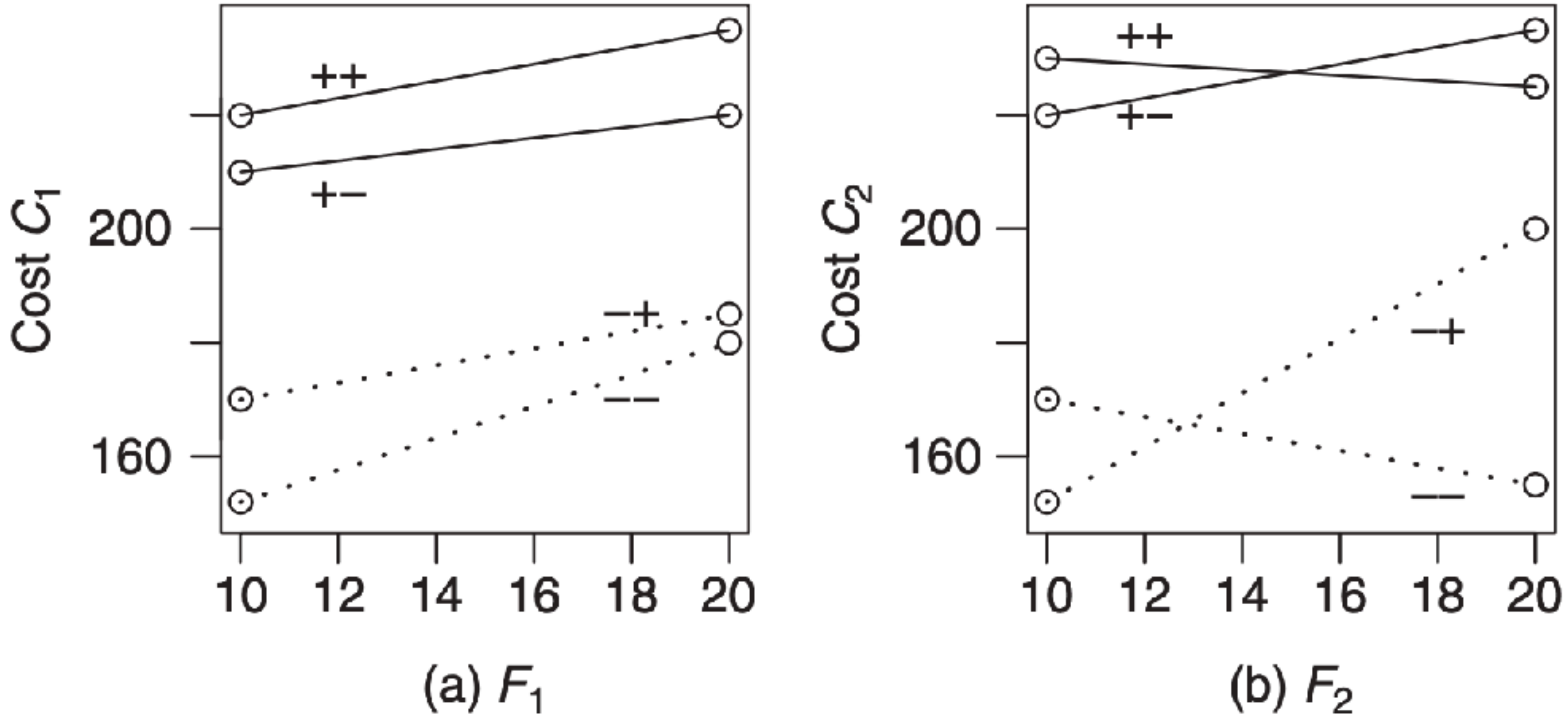


Figure 2.6. Main effects and interaction effects. Panel (a) shows three main effects from F_1, F_2 and F_3: C_1 increases by about the same amount when each factor changes from (-) to (+). Panel (b) shows an interaction effect: C_2 increases or decreases depending on whether F_2 matches F_3.

Experimental Design Basics

Some notes on full-factorial design and factor reduction

- Design points are exponential in number of factors, thus, huge space
- No need to evaluate independent factors with full factorization

To reduce the number of factors, consider:

- Merging similar factors
- Removing factors that can be inferred accurately
- Converting factors to noise parameters (some ratios instead of distinct factors)
- Limiting the scope of the experiments by fixing some factors
- Eliminating not-so-important factors

Design of Experiments - Summary

2.1 *Leverage the pilot study – and the literature – to create better workhorse experiments.*

2.2 *Never assume. Design experiments with built-in safeguards against bugs and artifacts, and be sure you can replicate your own results.*

2.3 *Experimental efficiency depends on the speed of the test program, the usability of the test environment, the quality of data returned, and the generality of conclusions drawn.*

2.4 *Choose input classes to support goals of correctness and generality, and to target the question at hand.*

2.5 *Choose as factors those parameters that are most important to performance, fix the parameters that are least relevant to performance, and let the other parameters vary.*

2.6 *When comparing algorithm (or program) design options, choose performance indicators and factors to highlight the differences among the options being compared.*

2.7 *Try a doubling experiment for a quick assessment of function growth.*

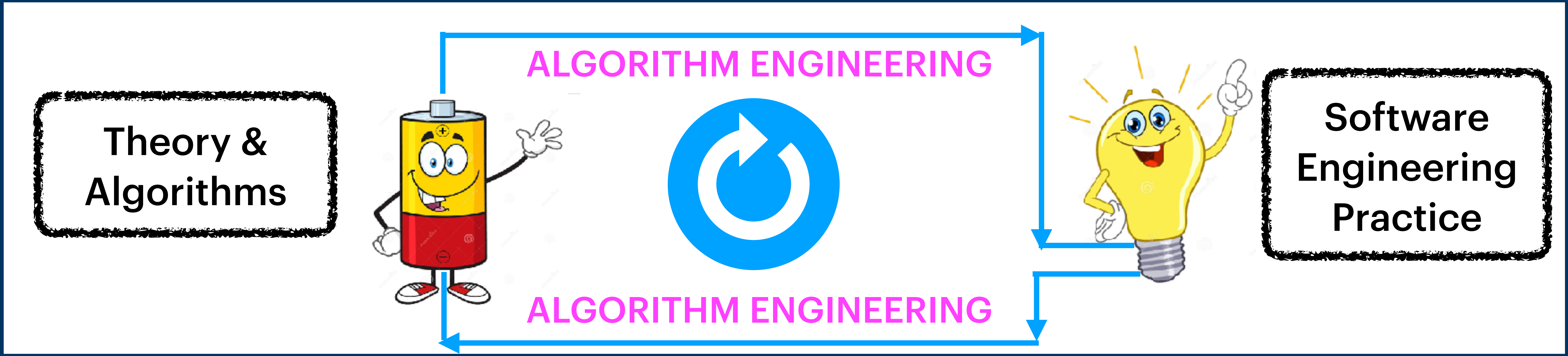
2.8 *The problem of analyzing a multidimensional function can be simplified by focusing on a small number of one-dimensional functions, ideally with similar shapes.*

2.9 *To study trends and functions, choose design points that exploit what you already know.*

2.10 *Full factorial designs maximize the information gained from one experiment.*

2.11 *When the experimental design is too big, apply factor-reduction strategies to reduce the size of the design with least damage to generality.*

NEXT LECTURE ...



ALGORITHM ENGINEERING

Lecture 4:
Design of Experiments - II

M. Oğuzhan Külekci - kulekci@itu.edu.tr