Recap of EC metaphor (1/2)

• A population of individuals exists in an environment with limited resources

• *Competition* for those resources causes selection of those *fitter* individuals that are better adapted to the environment

• These individuals act as seeds for the generation of new individuals through recombination and mutation

• The new individuals have their fitness evaluated and compete (possibly also with parents) for survival.

• Over time *Natural selection* causes a rise in the fitness of the population
Recap of EC metaphor (2/2)

- EAs fall into the category of “generate and test” algorithms
- They are stochastic, population-based algorithms
- Variation operators (recombination and mutation) create the necessary diversity and thereby facilitate novelty
- Selection reduces diversity and acts as a force pushing quality
Chapter 3: What is an Evolutionary Algorithm?

• Scheme of an EA
• Main EA components:
  – Representation / evaluation / population
  – Parent selection / survivor selection
  – Recombination / mutation
• Examples: eight-queens problem
• Typical EA behaviour
• EAs and global optimisation
• EC and neighbourhood search
Scheme of an EA: General scheme of EAs

- Initialisation
- Population
- Parent selection
- Parents
- Recombination (crossover)
- Mutation
- Survivor selection
- Offspring
- Termination
Scheme of an EA:
EA scheme in pseudo-code

BEGIN

INITIALISE population with random candidate solutions;
EVALUATE each candidate;
REPEAT UNTIL (TERMINATION CONDITION is satisfied) DO
  1 SELECT parents;
  2 RECOMBINE pairs of parents;
  3 MUTATE the resulting offspring;
  4 EVALUATE new candidates;
  5 SELECT individuals for the next generation;
OD
END
Scheme of an EA: Common model of evolutionary processes

- **Population** of individuals
- Individuals have a **fitness**
- **Variation** operators: crossover, mutation
- **Selection** towards higher fitness
  - “survival of the fittest” and
  - “mating of the fittest”

**Neo Darwinism:**
Evolutionary progress towards higher life forms = Optimization according to some fitness-criterion (optimization on a fitness landscape)

Now the theory of evolution incorporates Mendel's genetics into *Darwin's* framework; the combined theory was called "**neo-Darwinism.**" (Recently, that cumbersome term is being replaced by the simpler "**Darwinism**"). According to this paradigm, evolution is driven by chance.
Scheme of an EA: Two pillars of evolution

There are two competing forces

**Increasing population diversity** by genetic operators
- mutation
- recombination

Push towards **novelty**

**Decreasing population diversity** by selection
- of parents
- of survivors

Push towards **quality**
Main EA components: Representation (1/2)

- **Role**: provides code for candidate solutions that can be manipulated by variation operators
- **Leads to two levels of existence**
  - phenotype: object in original problem context, the outside
  - genotype: code to denote that object, the inside (chromosome, “digital DNA”)
- **Implies two mappings**:
  - Encoding: phenotype => genotype (not necessarily one to one)
  - Decoding: genotype => phenotype (must be one to one)
- **Chromosomes contain genes**, which are in (usually fixed) positions called loci (sing. locus) and have a value (allele)
Main EA components: Representation (2/2)

Example: represent integer values by their binary code

In order to find the global optimum, every feasible solution must be represented in genotype space.
Main EA components: Evaluation (fitness) function

• Role:
  – Represents the task to solve, the requirements to adapt to (can be seen as “the environment”)
  – Enables selection (provides basis for comparison)
  – e.g., some phenotypic traits are advantageous, desirable, e.g. big ears cool better, these traits are rewarded by more offspring that will expectedly carry the same trait

• A.k.a. *quality* function or *objective* function

• Assigns a single real-valued fitness to each phenotype which forms the basis for selection
  – So the more discrimination (different values) the better

• Typically we talk about fitness being maximised
  – Some problems may be best posed as minimisation problems, but conversion is trivial
Main EA components: Population (1/2)

- Role: holds the candidate solutions of the problem as individuals (genotypes)
- Formally, a population is a multiset of individuals, i.e. repetitions are possible
- Population is the basic unit of evolution, i.e., the population is evolving, not the individuals
- Selection operators act on population level
- Variation operators act on individual level
Main EA components: Population (2/2)

- Some sophisticated EAs also assert a spatial structure on the population e.g., a grid
- Selection operators usually take whole population into account i.e., reproductive probabilities are relative to current generation
- Diversity of a population refers to the number of different fitnesses / phenotypes / genotypes present (note: not the same thing)
Main EA components: Selection mechanism (1/3)

Role:

- Identifies individuals
  - to become parents
  - to survive
- Pushes population towards higher fitness
- Usually probabilistic
  - high quality solutions more likely to be selected than low quality
  - but not guaranteed
  - even worst in current population usually has non-zero probability of being selected
- This *stochastic* nature can aid escape from local optima
Main EA components: Selection mechanism (2/3)

Example: roulette wheel selection

fitness(A) = 3
fitness(B) = 1
fitness(C) = 2

In principle, any selection mechanism can be used for parent selection as well as for survivor selection.
Main EA components: Selection mechanism (3/3)

- Survivor selection A.k.a. **replacement**
- Most EAs use fixed population size so need a way of going from (parents + offspring) to next generation
- Often deterministic (while parent selection is usually stochastic)
  - Fitness based: e.g., rank parents + offspring and take best
  - Age based: make as many offspring as parents and delete all parents
- Sometimes a combination of stochastic and deterministic (elitism)
Main EA components: Variation operators

• Role: to generate new candidate solutions
• Usually divided into two types according to their \textit{arity} (number of inputs):
  – Arity 1: mutation operators
  – Arity >1: recombination operators
  – Arity = 2 typically called \textit{crossover}
  – Arity > 2 is formally possible, seldom used in EC
• There has been much debate about relative importance of recombination and mutation
  – Nowadays most EAs use both
  – Variation operators must match the given representation
Main EA components: Mutation (1/2)

- Role: causes small, random variance
- Acts on one genotype and delivers another
- Element of randomness is essential and differentiates it from other unary heuristic operators
- Importance ascribed depends on representation and historical dialect:
  - Binary GAs – background operator responsible for preserving and introducing diversity
  - EP for FSM’s / continuous variables – only search operator
  - GP – hardly used
- May guarantee connectedness of search space and hence convergence proofs
Main EA components: Mutation (2/2)

before  

```
1 1 1 1 1 1 1
```

after  

```
1 1 1 0 1 1 1
```
Main EA components: Recombination (1/2)

- Role: merges information from parents into offspring
- Choice of what information to merge is stochastic
- Most offspring may be worse, or the same as the parents
- Hope is that some are better by combining elements of genotypes that lead to good traits
- Principle has been used for millennia by breeders of plants and livestock
Main EA components: Recombination (2/2)

Parents

```
0 0 0 0 0 0 0
1 1 1 1 1 1
```

Offspring

```
0 0 0 1 1 1 1
1 1 1 0 0 0 0
```
Main EA components: Initialisation / Termination

- Initialisation usually done at random,
  - Need to ensure even spread and mixture of possible allele values
  - Can include existing solutions, or use problem-specific heuristics, to “seed” the population

- Termination condition checked every generation
  - Reaching some (known/hoped for) fitness
  - Reaching some maximum allowed number of generations
  - Reaching some minimum level of diversity
  - Reaching some specified number of generations without fitness improvement
Main EA components: What are the different types of EAs

- Historically different flavours of EAs have been associated with different data types to represent solutions
  - Binary strings : Genetic Algorithms
  - Real-valued vectors : Evolution Strategies
  - Finite state Machines: Evolutionary Programming
  - LISP trees: Genetic Programming

- These differences are largely irrelevant, best strategy
  - choose representation to suit problem
  - choose variation operators to suit representation

- Selection operators only use fitness and so are independent of representation
Example:
The 8-queens problem

Place 8 queens on an 8x8 chessboard in such a way that they cannot check each other
The 8-queens problem: Representation

**Phenotype:** a board configuration

**Genotype:** a permutation of the numbers 1–8
The 8-queens problem: Fitness evaluation

- **Penalty** of one queen: the number of queens she can check

- Penalty of a configuration: the sum of penalties of all queens

- **Note:** penalty is to be minimized

- **Fitness** of a configuration: inverse penalty to be maximized
The 8-queens problem: Mutation

Small variation in one permutation, e.g.:
• swapping values of two randomly chosen positions,

\[
\begin{array}{cccccccc}
1 & 3 & 5 & 2 & 6 & 4 & 7 & 8 \\
\end{array}
\quad \rightarrow \quad
\begin{array}{cccccccc}
1 & 3 & 7 & 2 & 6 & 4 & 5 & 8 \\
\end{array}
\]
The 8-queens problem: Recombination

Combining two permutations into two new permutations:
- choose random crossover point
- copy first parts into children
- create second part by inserting values from other parent:
  - in the order they appear there
  - beginning after crossover point
  - skipping values already in child

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

```
1 3 5 4 2 8 7 6
8 7 6 2 4 1 3 5
```
The 8-queens problem: Selection

- **Parent selection:**
  - Pick 5 parents and take best two to undergo crossover

- **Survivor selection (replacement):**
  - When inserting a new child into the population, choose an existing member to replace by:
    - sorting the whole population by decreasing fitness
    - enumerating this list from high to low
    - replacing the first with a fitness lower than the given child
The 8-queens problem: Summary

<table>
<thead>
<tr>
<th>Representation</th>
<th>Permutations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recombination</td>
<td>“Cut-and-crossfill” crossover</td>
</tr>
<tr>
<td>Recombination probability</td>
<td>100%</td>
</tr>
<tr>
<td>Mutation</td>
<td>Swap</td>
</tr>
<tr>
<td>Mutation probability</td>
<td>80%</td>
</tr>
<tr>
<td>Parent selection</td>
<td>Best 2 out of random 5</td>
</tr>
<tr>
<td>Survival selection</td>
<td>Replace worst</td>
</tr>
<tr>
<td>Population size</td>
<td>100</td>
</tr>
<tr>
<td>Number of Offspring</td>
<td>2</td>
</tr>
<tr>
<td>Initialisation</td>
<td>Random</td>
</tr>
<tr>
<td>Termination condition</td>
<td>Solution or 10,000 fitness evaluation</td>
</tr>
</tbody>
</table>

Note that is *only one possible* set of choices of operators and parameters
Typical EA behaviour: Stages

Stages in optimising on a 1-dimensional fitness landscape

Early stage: quasi-random population distribution

Mid-stage: population arranged around/on hills

Late stage: population concentrated on high hills
Typical EA behaviour: Working of an EA demo (1/2)

Searching a fitness landscape without “niching”
Typical EA behaviour: Working of an EA demo (2/2)

Searching a fitness landscape with “niching”
Typical EA behaviour: Typical run: progression of fitness

Typical run of an EA shows so-called “anytime behavior”
Typical EA behaviour: Are long runs beneficial?

- Answer:
  - It depends on how much you want the last bit of progress
  - May be better to do more short runs
Typical EA behaviour: Is it worth expending effort on smart initialisation?

- Answer: it depends.
  - Possibly good, if good solutions/methods exist.
  - Care is needed, see chapter/lecture on hybridisation.
Typical EA behaviour: Evolutionary Algorithms in context

- There are many views on the use of EAs as robust problem solving tools
- For most problems a problem-specific tool may:
  - perform better than a generic search algorithm on most instances,
  - have limited utility,
  - not do well on all instances
- Goal is to provide robust tools that provide:
  - evenly good performance
  - over a range of problems and instances
Typical EA behaviour: EAs as problem solvers: Goldberg view (1989)
Typical EA behaviour: EAs and domain knowledge

• Trend in the 90’s:
  adding problem specific knowledge to EAs
  (special variation operators, repair, etc)

• Result: EA performance curve “deformation”:
  – better on problems of the given type
  – worse on problems different from given type
  – amount of added knowledge is variable

• Recent theory suggests the search for an “all-purpose” algorithm may be fruitless
Typical EA behaviour:
EAs as problem solvers: Michalewicz view (1996)
EC and global optimisation

• Global Optimisation: search for finding best solution $x^*$ out of some fixed set $S$
• Deterministic approaches
  – e.g. box decomposition (branch and bound etc)
  – Guarantee to find $x^*$,
  – May have bounds on runtime, usually super-polynomial
• Heuristic Approaches (generate and test)
  – rules for deciding which $x \in S$ to generate next
  – no guarantees that best solutions found are globally optimal
  – no bounds on runtime

• “I don’t care if it works as long as it converges”
  vs.
• “I don’t care if it converges as long as it works”
EC and neighbourhood search

- Many heuristics impose a neighbourhood structure on $S$
- Such heuristics may guarantee that best point found is *locally optimal* e.g. Hill-Climbers:
  - But problems often exhibit many local optima
  - Often very quick to identify good solutions
- EAs are distinguished by:
  - Use of population,
  - Use of multiple, stochastic search operators
  - Especially variation operators with arity $>1$
  - Stochastic selection

- Question: what is the neighbourhood in an EA?