COMP6925
APPLIED OPERATIONS RESEARCH

Lab 2

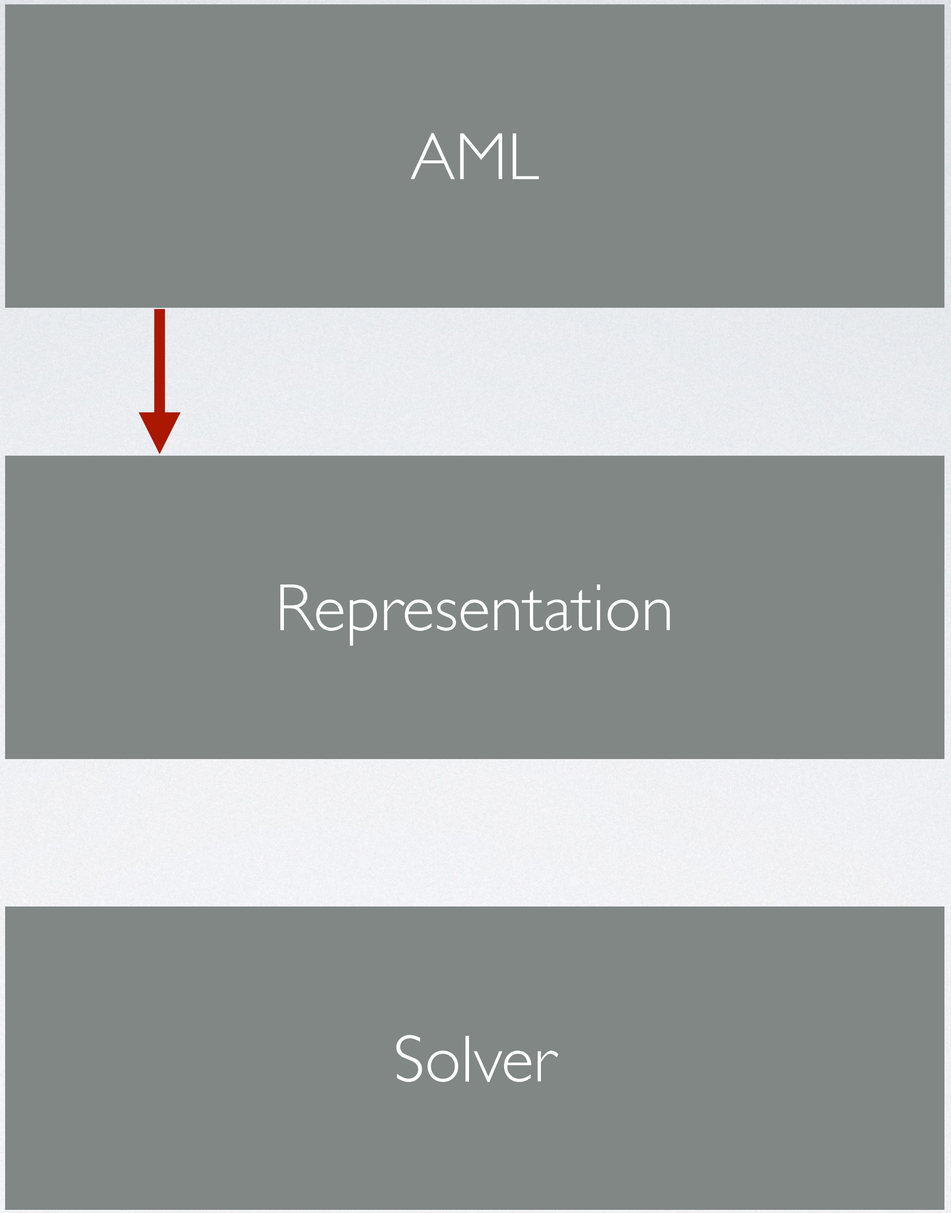
SOFTWARE AND OPTIMISATION

- Myriad of optimisation scenarios
  - A wide taxonomy of problems
  - Algorithms/methods developed for different classes of problems
  - Don’t want to re-implement algorithms for each instance
  - Abstracted into solvers that accept problem specs as input and outputs results in appropriate format
  - Do we have to become conversant with the particulars of many solvers?

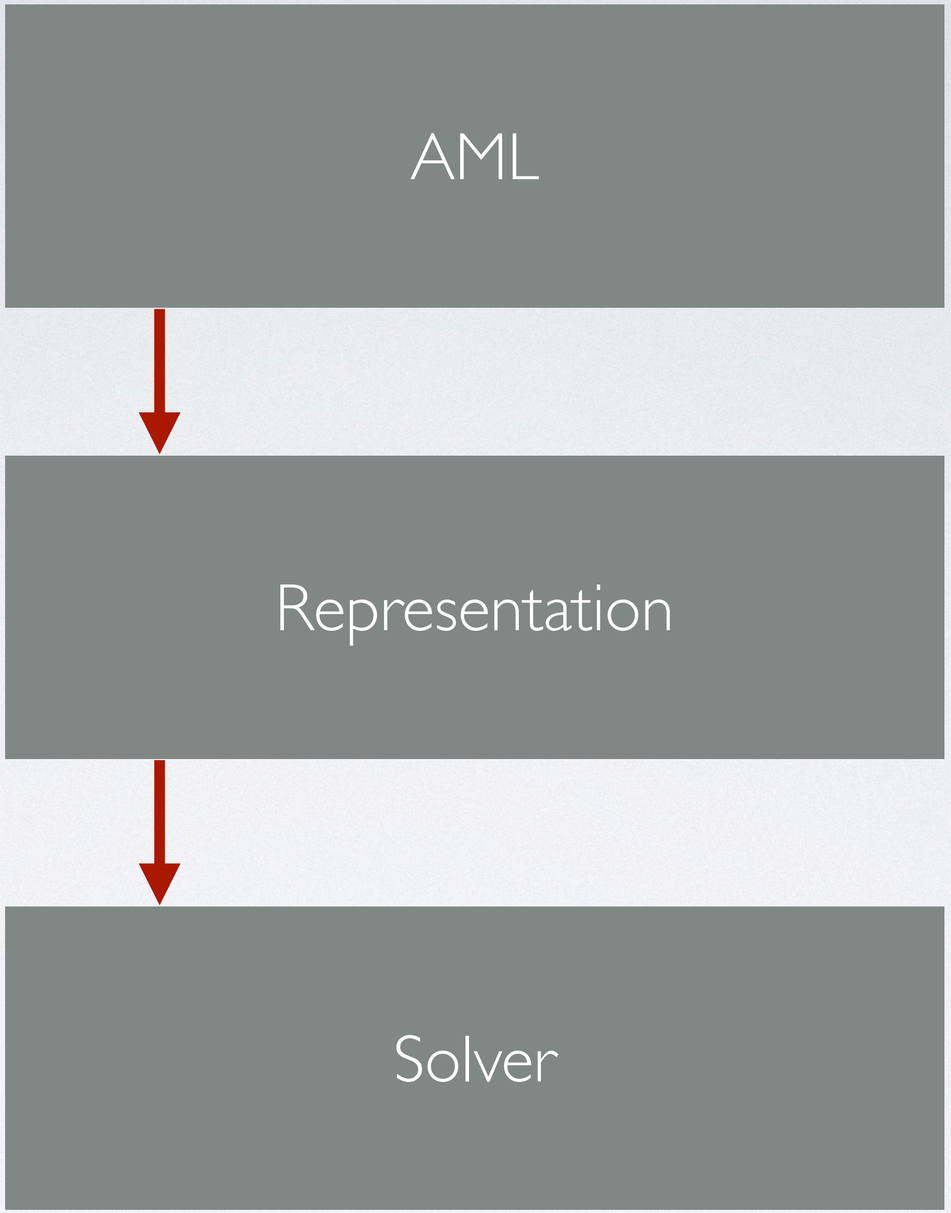
ALGEBRAIC MODELLING LANGUAGES

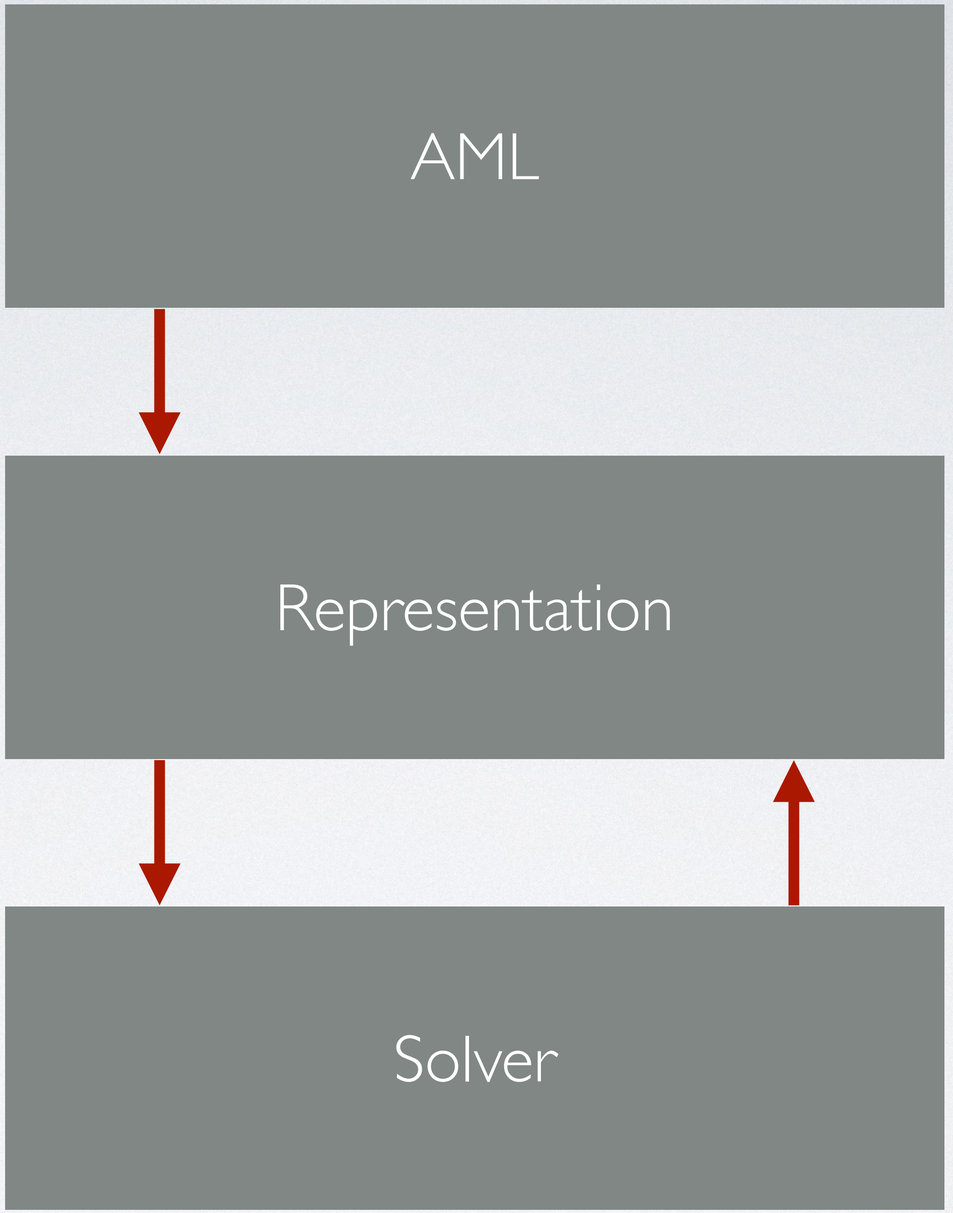
- Many problems share same components:
  - Objective
  - Constraints
- AMLs provide interface to specify many optimisation problems in terms of these components
- Provides algebra for encoding optimisation problems
- Unified interface

AML translates into representation expected by solver

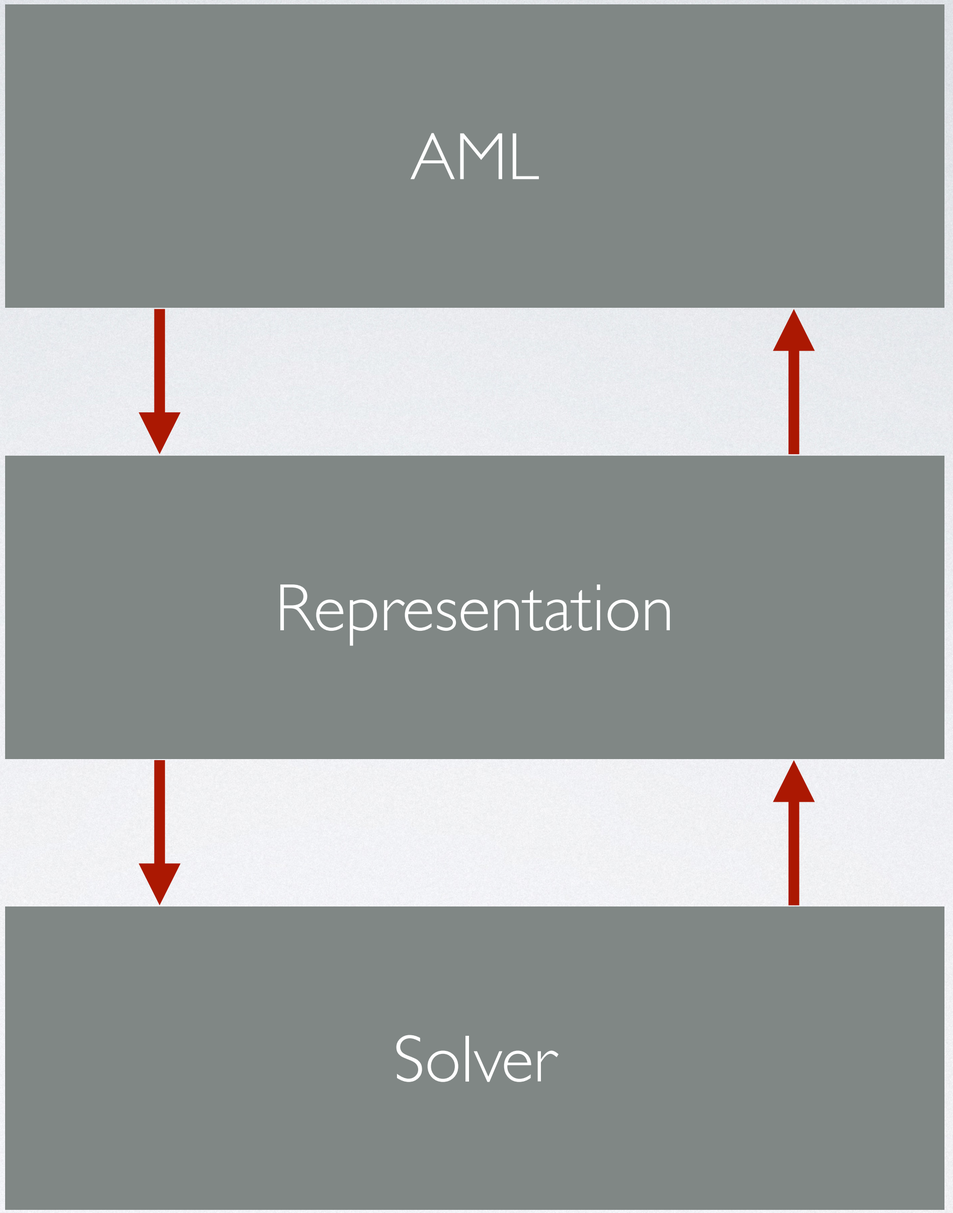


Representation is fed into the solver





Solver executes algorithm on input and returns result in solver specific format



Output Representation is then adapted into interface for access of results

EMBEDDED ALEGBRAIC MODELLING LANGUAGES

- Several independent AMLs - standalone
- Other AMLs are embedding as library in another programming language
- In this course we will use JuMP - an AML embeddded in Julia
    - Might also look into Convex.jl

```
using Pkg
Pkg.add("JuMP") # to install JuMP library and its dependencies
Pkg.add("Cbc") # solver to solver linear programming problems
```

JUMP

- JuMP interacts with solvers by passing through the MathOptInterface library
- Makes use of macros to construct optimisation models
- Uses function calls to initiate solving process and to extract results

EXAMPLES

Ralph Edmund loves steaks and potatoes. Therefore, he has decided to go on a steady diet of only these two foods (plus some liquids and vitamin supplements) for all his meals. Ralph realises that this isn't the healthiest diet, so he wants to make sure that he eats the right quantities of the two foods to satisfy some key nutritional requirements. He has obtained the nutritional and cost information shown at the top of the next column.

Ralph wishes to determine the number of daily servings (may be fractional) of steak and potatoes that will meet these requirements at a minimum cost.

Ingredient	Grams of Ingredient per Serving		Daily Requirement (Grams)
	Steak	Potatoes	
Carbohydrates	5	15	≥ 50
Protein	20	5	≥ 40
Fat	15	2	≤ 60
Cost per serving	\$8	\$4	

```julia
using JuMP
using Clp

CARBS = 50
PROTEIN = 40
FAT = 60

m = Model(Clp.Optimizer)

@variable(m, steak ≥ 0)
@variable(m, potatoes ≥ 0)

@objective(m, Min, 8 * steak + 4 * potatoes)

@constraint(m, 5 * steak + 15 * potatoes ≥ CARBS)
@constraint(m, 20 * steak + 5 * potatoes ≥ PROTEIN)
@constraint(m, 15 * steak + 2 * potatoes ≤ FAT)

print(m)

optimize!(m)

status = termination_status(m)

println("Solution status: ", status)

println("Objective value: ", objective_value(m))
println("potatoes = ", value(steak))
println("steak = ", value(potatoes))
```

```julia
using JuMP
using Clp

PROPORTION_MAT = [
    [5.0 15.0]
    [20.0 5.0]
    [-15.0 -2.0]
]

LIMITS = [50.0, 40.0, -60.0]

COSTS = [8.0, 4.0]

m = Model(Clp.Optimizer)
@variable(m, x[1:2] >= 0)
@objective(m, Min, COSTS' * x)
@constraint(m, PROPORTION_MAT * x .≥ LIMITS)

print(m)

optimize!(m)

status = termination_status(m)

println("Solution status: ", status)

println("Objective value: ", objective_value(m))

println("Values: ", value.(x))
```

Larry Edison is the director of the Computer Center for Buckly College. He now needs to schedule the staffing of the center. It is open from 8 A.M. until midnight. Larry has monitored the usage of the center at various times of the day, and determined that the following number of computer consultants are required:

Time of Day	Minimum Number of Consultants Required to Be on Duty
8 A.M.–noon	4
Noon–4 P.M.	8
4 P.M.–8 P.M.	10
8 P.M.–midnight	6

Two types of computer consultants can be hired: full-time and part-time. The full-time consultants work for 8 consecutive hours in any of the following shifts: morning (8 A.M.–4 P.M.), afternoon (noon–8 P.M.), and evening (4 P.M.–midnight). Full-time consultants are paid $40 per hour. Part-time consultants can be hired to work any of the four shifts listed in the above table. Part-time consultants are paid $30 per hour.

An additional requirement is that during every time period, there must be at least 2 full-time consultants on duty for every part- time consultant on duty.

Larry would like to determine how many full-time and how many part-time workers should work each shift to meet the above requirements at the minimum possible cost.

A state wants to plan its electricity capacity for the next T years. The state has a forecast of d_t megawatts, presumed accurate, of the demand for electricity during year $t = 1, \ldots, T$. The existing capacity, which is in oil-fired plants, that will not be retired and will be available during year t, is e_t. There are two alternatives for expanding electric capacity: coal-fired or nuclear power plants. There is a capital cost of c_t per megawatt of coal-fired capacity that becomes operational at the beginning of year t. The corresponding capital cost for nuclear power plants is n_t. For various political and safety reasons, it has been decided that no more than 20% of the total capacity should ever be nuclear. Coal plants last for 20 years, while nuclear plants last for 15 years. A least cost capacity expansion plan is desired.

The first step in formulating this problem as a linear programming problem is to define the decision variables. Let x_t and y_t be the amount of coal (respectively, nuclear) capacity brought on line at the beginning of year t. Let w_t and z_t be the total coal (respectively, nuclear) capacity available in year t. The cost of a capacity expansion plan is therefore,

```julia
using JuMP
using Clp

DEMANDS = []
EXCESS = []
NUC_COST = []
COAL_COST = []
T = length(DEMANDS)

m = Model(Clp.Optimizer)
@variable(m, x[1:T] >= 0)
@variable(m, y[1:T] >= 0)
@variable(m, w[1:T] >= 0)
@variable(m, z[1:T] >= 0)
@objective(m, Min, sum(NUC_COST[i] * y[i] + COAL_COST[i] * x[i] for i = 1:T))

for t = 1:T
    @constraint(m, w[t] == sum(x[t:-1:max(1, t-19)]))
    @constraint(m, z[t] == sum(y[t:-1:max(1, t-14)]))
end

@constraint(m, w + z + EXCESS .≥ DEMANDS)
@constraint(m, 0.8 * z - 0.2 * w .≤ 0.2 * EXCESS)

println(m)

optimize!(m)

status = termination_status(m)

println("Solution status: ", status)

println("Objective value: ", objective_value(m))
println("x = ", value.(x))
println("y = ", value.(y))
```

Larry Edison is the director of the Computer Center for Buckly College. He now needs to schedule the staffing of the cen- ter. It is open from 8 A.M. until midnight. Larry has monitored the usage of the center at various times of the day, and determined that the following number of computer consultants are required: