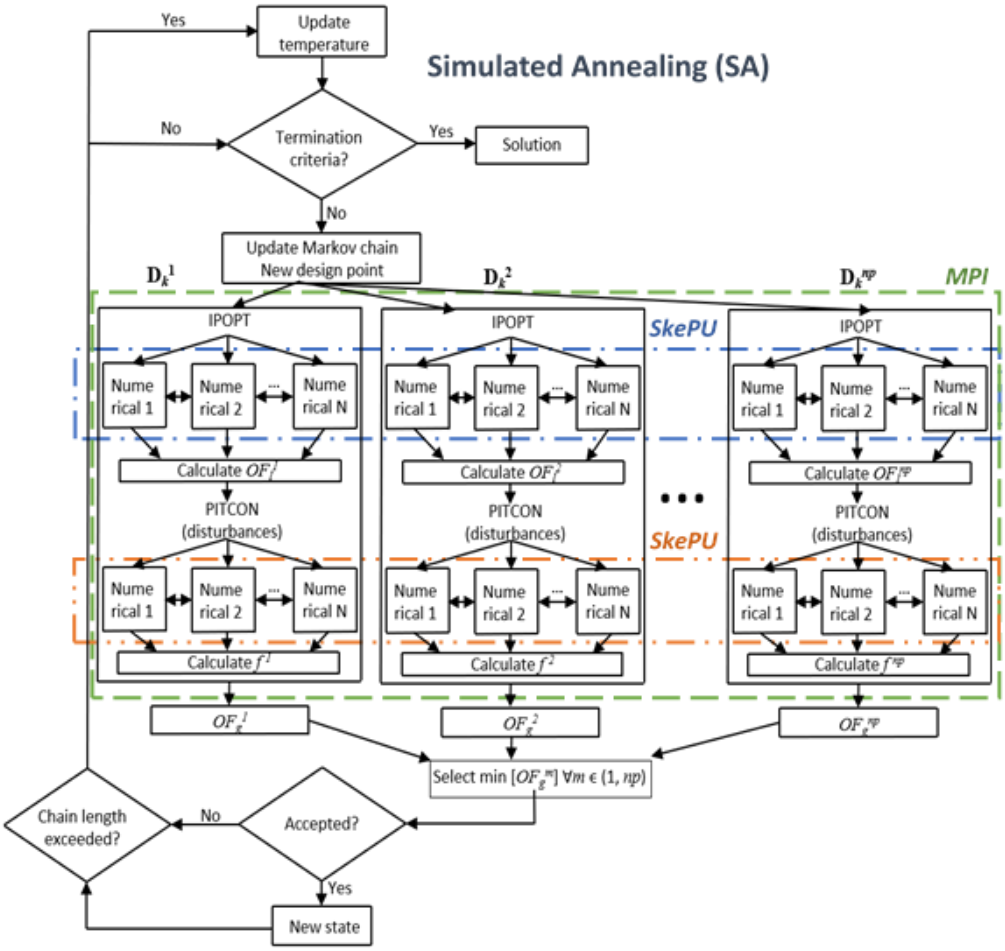
# EXA2PRO SkePU tool applied to an application for designing materials and processes for $CO_2$ capture

## A. Short Application description:

The $CO_2$ capture case study involves the use of an advanced optimization approach for the simultaneous design and control of chemical processes and materials used in $CO_2$ capture. The problem involves large combinatorial complexity due to the very large number of discrete and continuous parameter combinations that need to be considered prior to the identification of an optimum solution. The consideration of disturbances imposes additional effort as they represent additional operating scenarios that must be investigated.




The optimization problem is approached through a hybrid algorithmic scheme which combines a stochastic search algorithm, called Simulated Annealing, a deterministic optimization algorithm, called IPOPT, and a homotopy-continuation algorithm, called PITCON.

## B. Applying EXA2PRO framework to CO2 capture (SkePU tool):

The Map skeleton uses an input collection, to produce an output element for each one in the input collection, using a user-written function. Therefore, the **Map skeleton** applies.

We developed three functions that use the **Map skeleton**. Within each step of the algorithmic scheme, the user-written functions evaluate the $CO_2$ process constraints at multiple points during the optimization. The constraints

evaluation implies the simulation of the model at the current state (design parameters, enthalpy, pressure, interpolation coefficients, etc.)

Function signature:

```
Modules mapFunc(Index1D idx, Vec<Variables> v, Vec<Enthalpy> e,…);
```

Usage:

```
auto RunnableKernel = Map(mapFunc);
Vector<Variables> v; Vector<Enthalpy> e; Vector<Modules> outVec;
RunnableKernel(outVec, v, e,…);
```

Sequential implementation:

```
for each module i:
  … CO2 process simulation …
  Compute constraints[i]
```

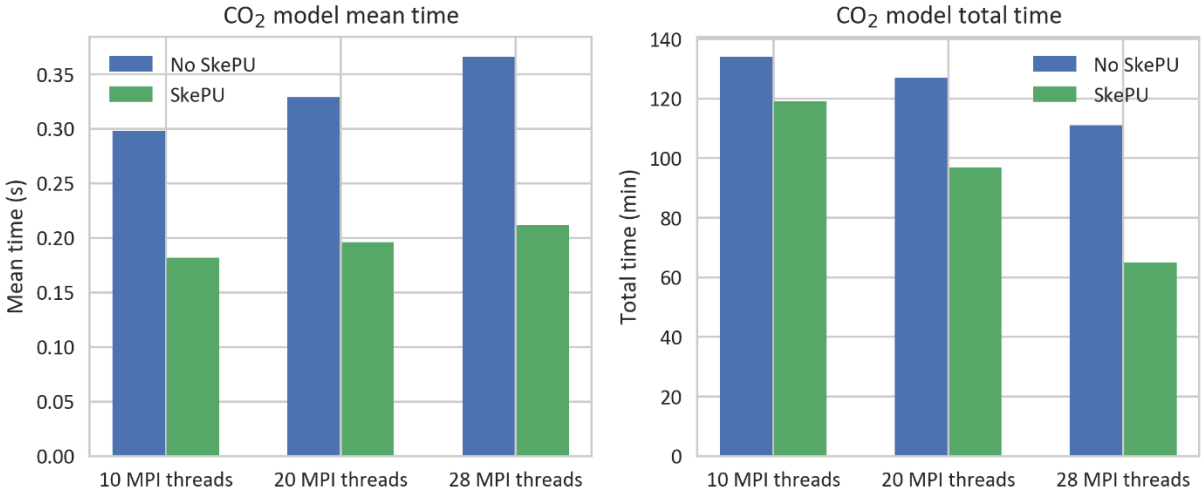
SkePU implementation:

User function

```
Function get_constraints(i, variables, enthalpy,…):
  … CO2 process simulation …
  Return constraints for module i

Kernel = Map(get_constraints)
constraints = Kernel(variables, enthalpy,…)
```

## C. Results




The tests were conducted on 2x Intel Xeon Gold 5120 2.2G, 28 physical cores (56 threads) with 64 GB RAM on GNU/Linux (Ubuntu 18.04 x86-64). A mixed parallel strategy was applied, using MPI at the Annealing (outer) level and the EXA2PRO framework (SkePU) at the $CO_2$ model (inner) level using OpenMP backend. Results are obtained for 1,000 Annealing iterations and show savings up to 41 % of the total time by using the SkePU version of $CO_2$ model compared to the original sequential implementation.

## D. Conclusions

Applying the EXA2PRO framework to the $CO_2$ capture application, enabled the porting of the application to CPU and GPU accelerators. The approach followed does not aim to improve a single time-consuming operation but rather to reduce the effort required for execution of multiple, repeated operations. This technique can be used to any optimization problem where the function evaluations are expensive to compute to improve the performance of the application.

**More Information:** N. Vasilas, P. Natsiavas, A. I. Papadopoulos, P. Seferlis, “Process Synthesis and Controllability Assessment of CO2 Capture Plants in a Parallel Environment”, PRES 2019.