Unit A2.3 Modeling Paradigms

Kenneth D. Forbus
Qualitative Reasoning Group
Northwestern University

Overview

- Compositional Modeling
- Perspectives
  - Multiple Ontologies
    - Example: Liquids
- Behavior, function, and teleology
    - Example: Teleological reasoning about thermodynamic cycles

Problems in building models

- Curse of fidelity
  - Level of detail and precision varies with task
  - Model too simple ⇒ inaccurate results
  - Model too complex ⇒ high costs to get data, wasted computational effort
- Clash of perspectives
  - Different problems require different perspectives
    - Container versus infinite source/sink
    - When to ignore thermal properties, electrical, vibration…
  - Choosing appropriate perspective can be hard
  - Conflicting alternatives must peacefully coexist

Compositional Modeling: Basics

- Explicit *modeling assumptions* included in domain theory
  - de Kleer & Brown’s *class-wide assumptions* informally captured some of this idea, but were never implemented
  - Organize modeling assumptions into *assumption classes*
  - Explicitly represent constraints between modeling assumptions
- *Model formulation algorithm* creates model
  - Inputs: Domain theory + scenario structural description + query + other stuff
  - Output: A model for the scenario appropriate for answering the query

CONSIDER assumptions

- Format: **`(consider <`*`specifier`*`>)`**
- Guides instantiation of model fragments
- Method 1: Explicit inclusion in model fragment definition
  - e.g., (consider (liquid can)) in :constraints of :participants of contained-liquid model fragment
- Method 2: Separate statements in domain theory
  - Satisfying participants necessary, but not sufficient, for instantiation of a model fragment
  - Two-pass process: Propose instantiations, accept/reject them

```
(defprocess (fluid-flow ?src-cs ?dst ?path)
  Participants ((?path :type fluid-path
                 :conditions (possible-path-state ?path ?st)
                             (connects-to ?path ?src ?dst))
             (?src-cs :type contained-stuff
               :form (C-S ?sub ?st ?src)
                        :conditions (Filled ?path ?src-cs))
             (?dst :type container)
             (?pr-src :conditions
                      (Pressure-Definer ?path ?src ?pr-src))
             (?pr-dst :conditions
                      (Pressure-Definer ?path ?dst ?pr-dst)))
  Conditions ((aligned ?path)
              (> (pressure ?pr-src :ABSOLUTE)
                 (pressure ?pr-dst :ABSOLUTE)))
 Consequences ((Quantity flow-rate)
        (Material-Flow ?sub ?st ?src ?dst ?path flow-rate)
        (Flow-Thru ?src-cs ?path)
        (I+ (Amount-of-in ?sub ?st ?dst) (A flow-rate))
        (I- (Amount-of-in ?sub ?st ?src) (A flow-rate))))
```

```
(defmodelFragment (simple-fluid-rate ?pi)
  :participants ((?pi :type (process-instance fluid-flow))
                 (?src :type contained-fluid
                       :conditions (src-of ?pi ?src))
                 (?dst :type contained-fluid
                       :conditions (dst-of ?pi ?dst))
                (?path :type fluid-path
                        :conditions (path-of ?pi?path)
                      (not (Consider
                             (fluid-conductance ?path)))))
 :conditions ((active ?pi))
 :consequences ((Q= (flow-rate ?pi)
                    (Q- (pressure ?src :ABSOLUTE)
                        (pressure ? dst :ABSOLUTE)))))
```

```
(defmodelfragment (variable-fluid-rate ?pi)
   :participants ((?pi :type (process-instance fluid-flow))
                 (?src :type contained-fluid
                       :conditions (src-of ?pi ?src))
                 (?dst :type contained-fluid
                       :conditions (dst-of ?pi ?dst))
                     (?path :type fluid-path
                        :conditions (path-of ?pi?path)
                         (Consider
                          (fluid-conductance ?path)))))
  :conditions ((active ?pi))
  :consequences ((Quantity (pressure ?src ?dst))
   (Q= (pressure ?src ?dst)
            (Q- (pressure ?src :ABSOLUTE)
               (pressure ?dst :ABSOLUTE)))
   (Q= (flow-rate ?pi)
       (*0+ (pressure ?src ?dst)
                    (fluid-conductance ?path)))))
```

Coherence

- Coherence enforced by explicit constraints between CONSIDER statements

  -
    ```
    (implies (consider thermal-properties)
        (forall ?st
         (implies (contained-stuff ?st)
            (consider
              (thermal-properties ?st)))))
    ```

  -
    ```
    (forall (?sub ?can)
      (implies (consider (thermal-in ?sub ?can))
        (forall ?st
         (implies (state ?st)
    ```

  -
    ```
            (consider (thermal-properties
                       (C-S ?sub ?st ?can)))))))
    ```

Assumption Classes

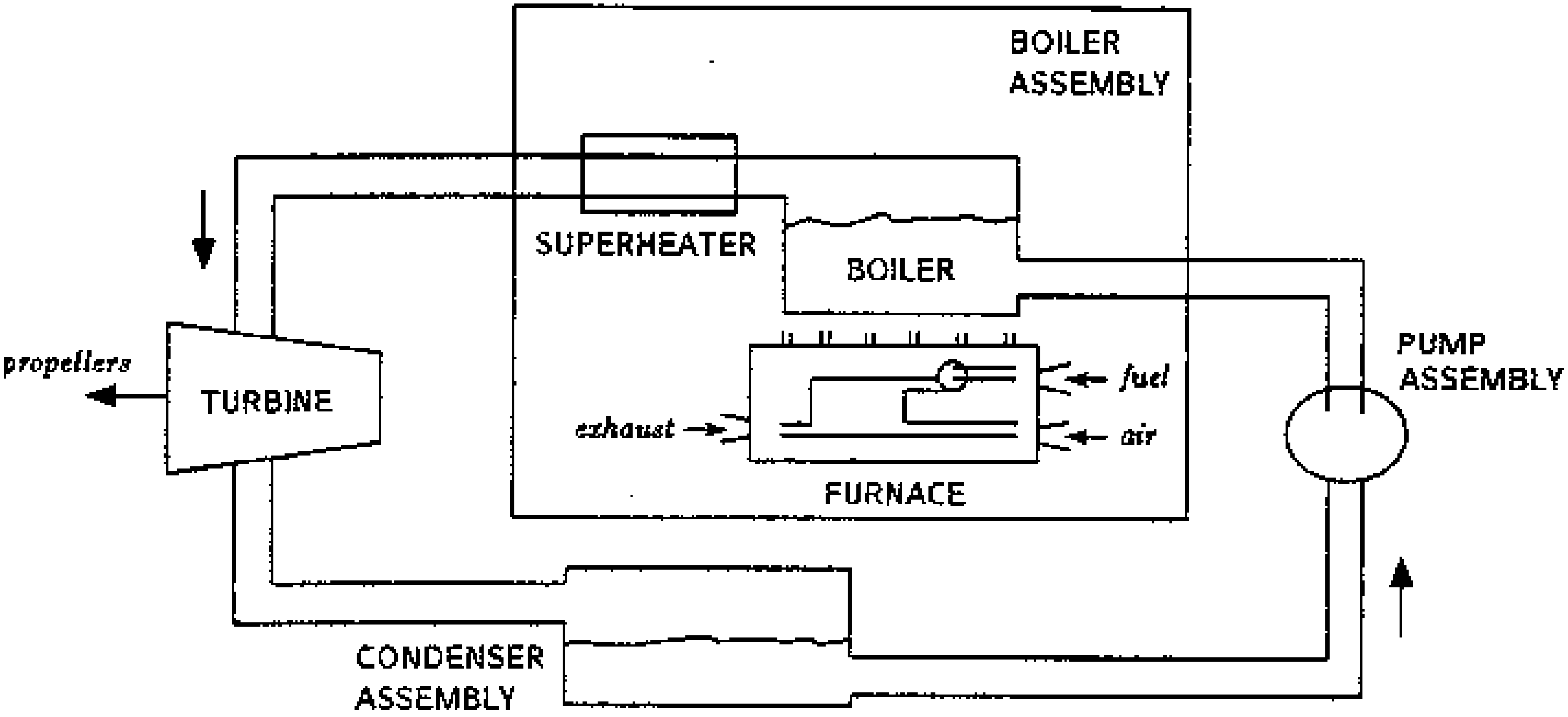
- = mutually exclusive, collectively exhaustive set of modeling alternatives
- A choice from every valid assumption class must be included for a model to be coherent
- Example:

```
(implies (thermodynamic-cycle ?cycle)
  (assumption-class (heat-engine ?cycle)
                    (refrigerator ?cycle)
                    (heat-pump ?cycle)))
```

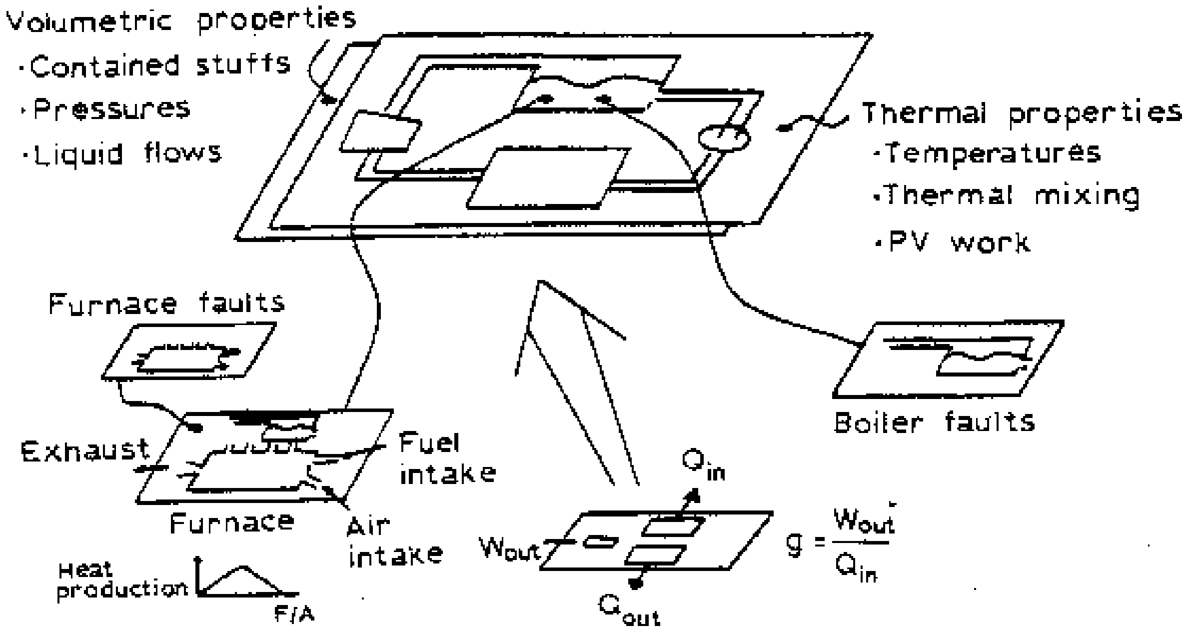
Operating Assumptions

- Constraints on system behavior that limit possibilities
- Examples:
  - Steady-state
  - No faults/failures
  - No high-frequency radiation effects
  - No thermal effects
- Effect: Greatly limit amount of analysis work

A simple steam plant



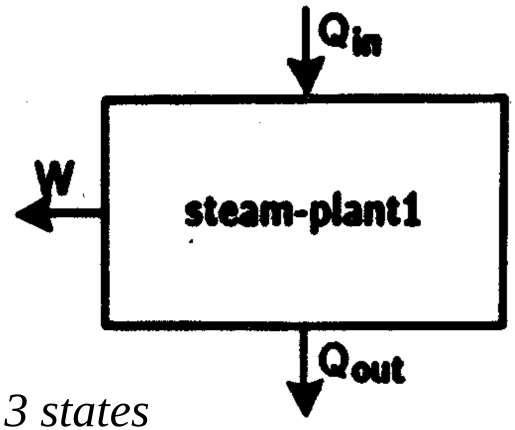
Qualitative model of the steam plant



- Domain theory
  - 8 object types, 37 model fragments (including 14 processes)
  - 1566 axiom-equivalents (horn clauses)
  - Comparison: Typical domain theory ~300
- Scenario model (complete)
  - 76 model fragment instances (including 21 processes), 79 quantities
  - 8617 horn clauses in ATMS
  - No computer ever survived through an envisionment

Q: What affects the efficiency of the plant?

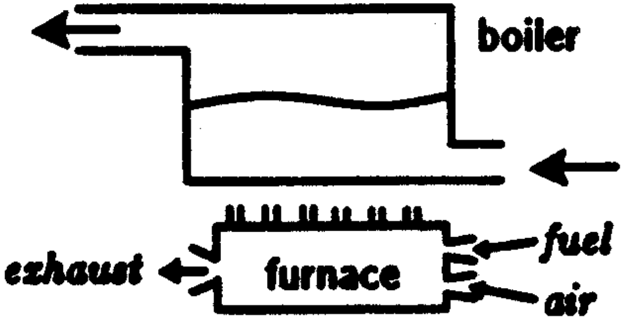
A: The efficiency of the plant is affected positively by the work rate of the turbine (W). It is also affected negatively by the energy input to the plant (Qin).



15 quantities, 41 ordinals, 6 model fragments (3 processes), 3 states

Q: How does the furnace's fuel/air ratio affect the boiler's steam production?

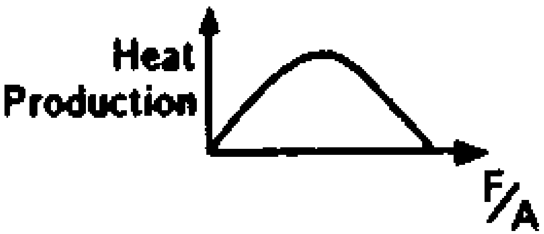
A: When the fuel/air ratio is below peak efficiency, an increase in the fuel/air ratio causes an increase in the boiler's heat rate, an increase in the boiler's heat rate causes an increase in the amount of steam in the boiler.



15 quantities, 41 ordinals, 6 model fragments (3 processes), 3 states

Q: What is causing black smoke to rise from the furnace?

A: Black smoke is rising from the furnace because the fuel/air ratio is greater thar the F/A saturation point for the furnace.



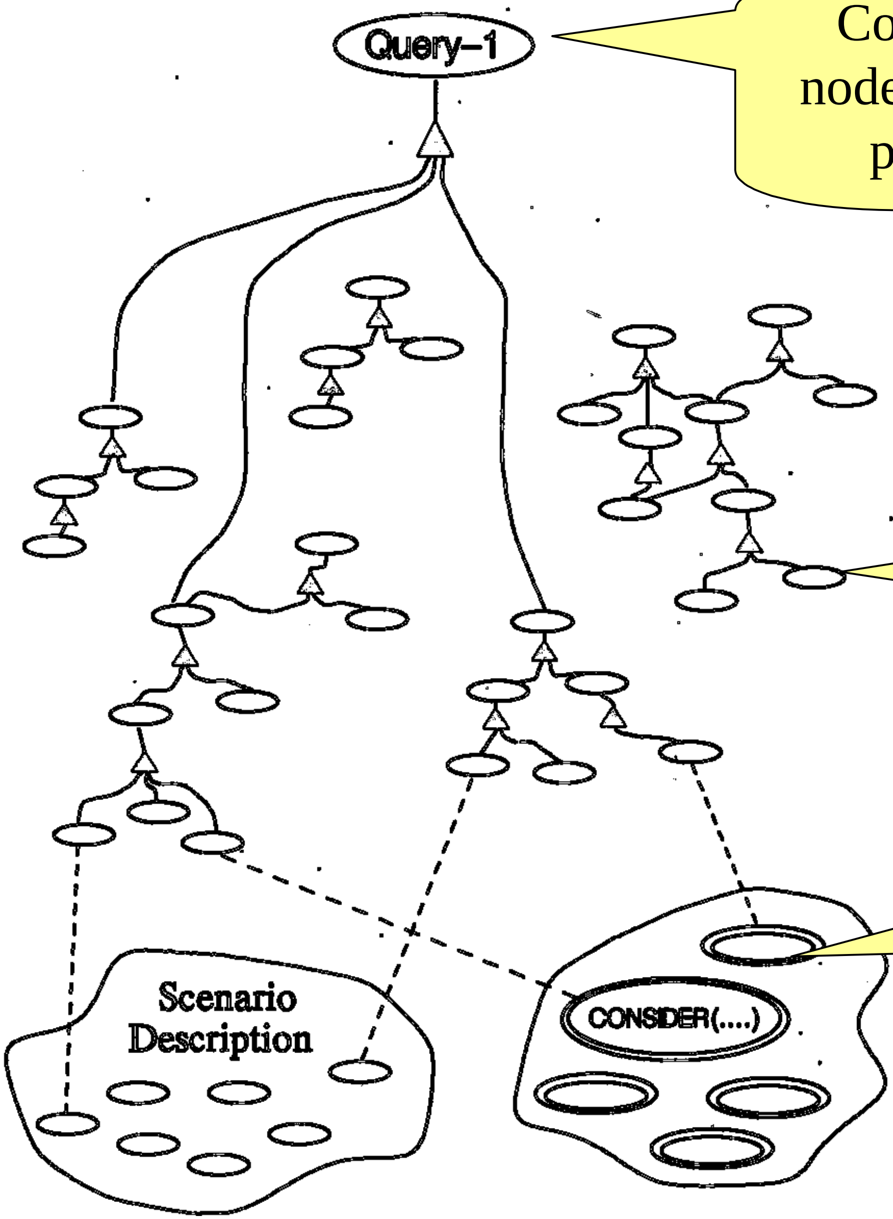
15 quantities, 41 ordinals, 6 model fragments (3 processes), 3 states

Problem: Given a query Q, a domain theory,
and a structural description of a system,
formulate the simplest model that will answer Q

Model Formulation Algorithm

- Instantiate all model fragments that match, ignoring modeling assumptions
- Find all combinations E of modeling assumptions that lead to models containing Q
  - This is straightforward with an ATMS
- Select $E_{min} \in \{E_i\}$ with fewest modeling assumptions
  - Heuristic: Fewer positive assumptions ⇒ simpler model
- Instantiate again, but under the logical environment E_{min}, respecting modeling assumptions

the ATMS model formulation algorithm



Consistent combinations arriving at node corresponding to query constitute possible modeling environments

Dependency structure relates modeling assumptions to terms in model

Modeling assumptions propagate through network, pruned by nogoods

Using system boundaries

- Many physical systems can be analyzed into subsystems
- Use system boundaries to help ensure coherence
  - Select uniform level of detail, same perspectives for all the components in the specific subsystem of interest
  - Can express this via axioms that propagate CONSIDER assumptions about phenomena through the parts of a system.
- Use system boundaries to avoid irrelevant detail
  - Systems above level of focus aren’t included
  - Systems below level of focus are replaced by “black box” functional equivalents

Efficiency of model formulation

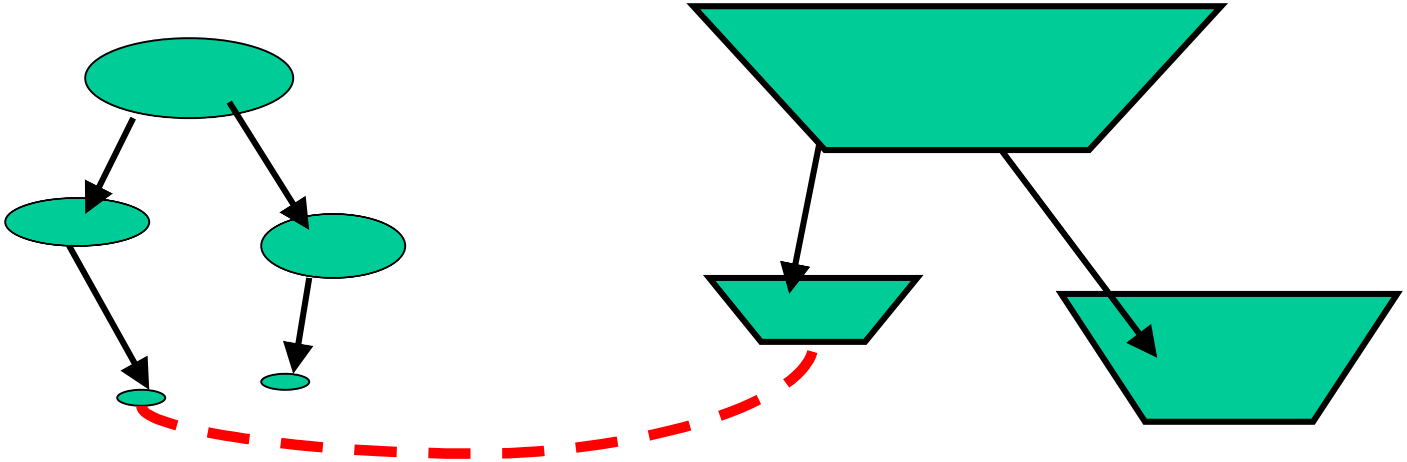
- Worst case exponential
  - Assumption classes ≡ choice sets
  - Model = consistent set of choices, simplest under some metric
  - Equivalent to P-SAT
- Observation: Human modelers are faster than this suggests.
- Question: Why?

Answer 1: They’re experienced

- Falkenhainer: Use analogy in modeling
  - Use modeling assumptions that worked in previous similar situations
  - Be on the lookout for problems like those you’ve encountered before
- Standardization within cultures
  - Engineering communities have agreed-upon guidelines about what modeling assumptions are appropriate.
    - Sometimes tacit, sometimes explicit
  - Educators have agreed-upon levels of explanation for phenomena to be taught

Answer 2: Restrict the problem

- Weaken optimality: *a* simplest model versus *the* simplest model
- Impose additional structure
  - Simplicity ordering within an assumption class
  - Limit interactions between assumption classes
- Can get polynomial-time model formulation

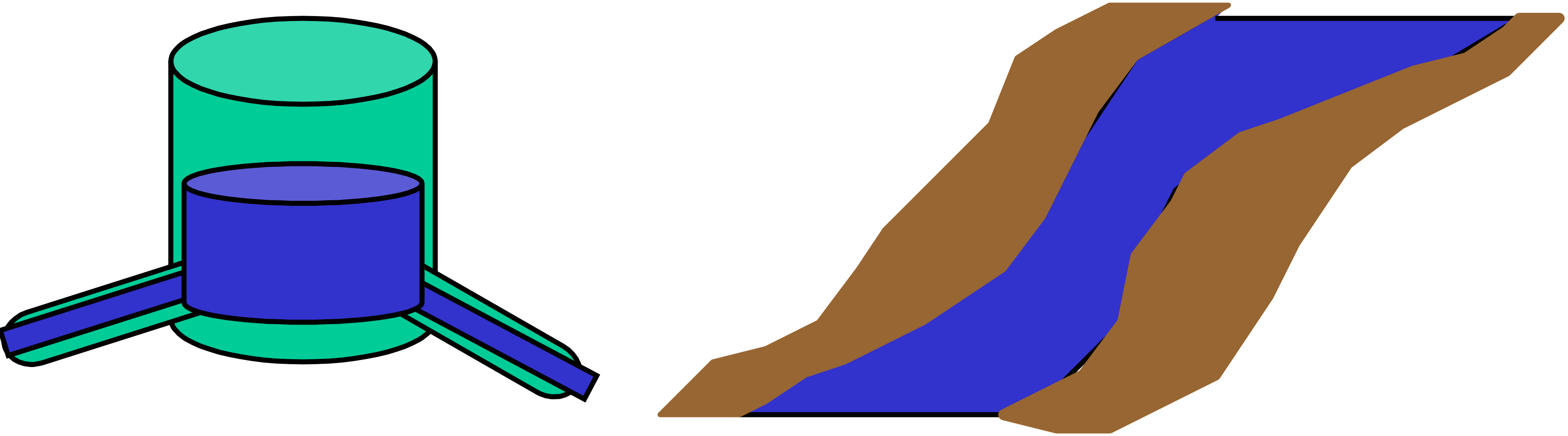


Time scales matter

- Physical phenomena occur at different timescales
  - Microseconds to millennia
- Can radically simplify relevance decisions
  - Slower phenomena can be ignored
  - Faster phenomena can be approximated by functional descriptions
  - Provides powerful pruning constraint for establishing model boundaries
- cf. papers by Iwasaki, Kuipers, Rickel, Yip

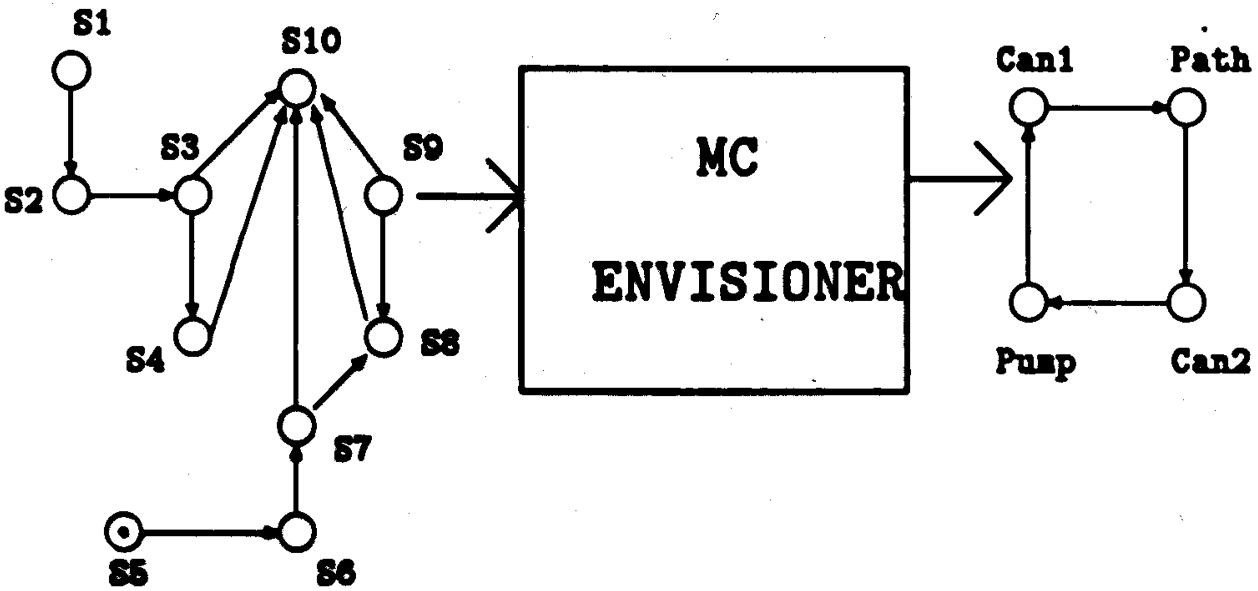
Multiple Perspectives: An example

- How to reason about liquids?
- Two models, due to Hayes
  - Contained stuff ontology: Individuate liquid via the space that it is in.
  - Piece of stuff ontology: Individuate liquid as a particular collection of molecules.

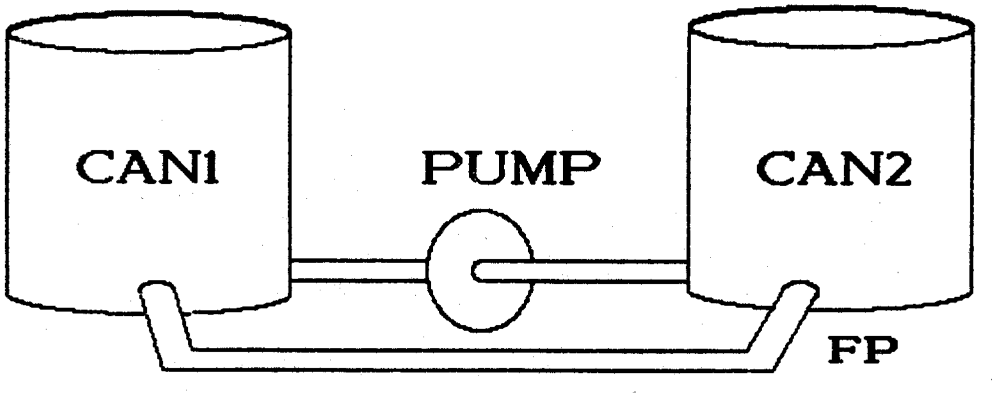


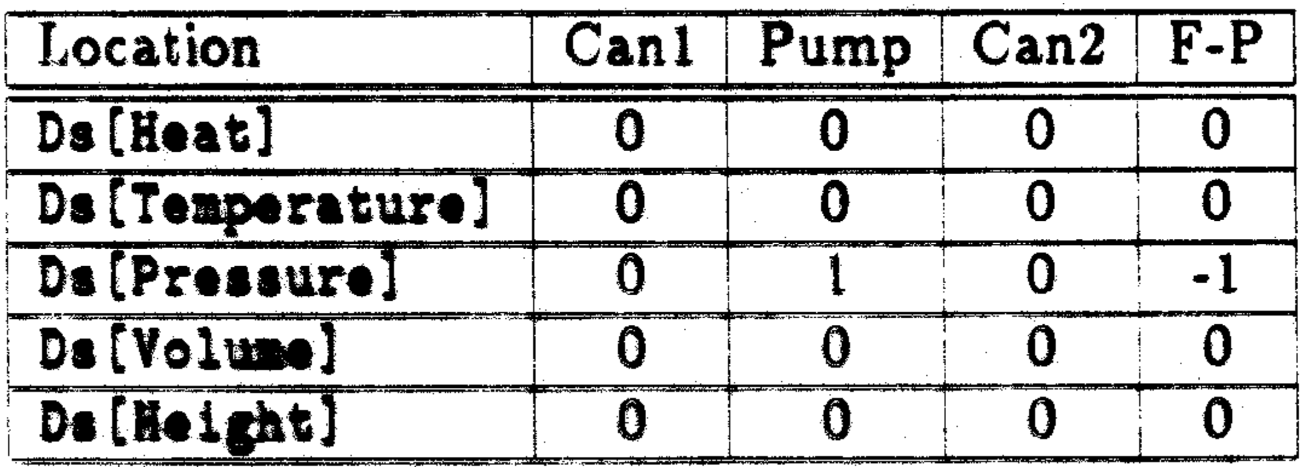
Molecular Collection ontology

- Idea: Follow a little piece of stuff around a system
  - So small that when it reaches a junction, it never splits apart
- Provides the perspective gained by tracing through a system of changes



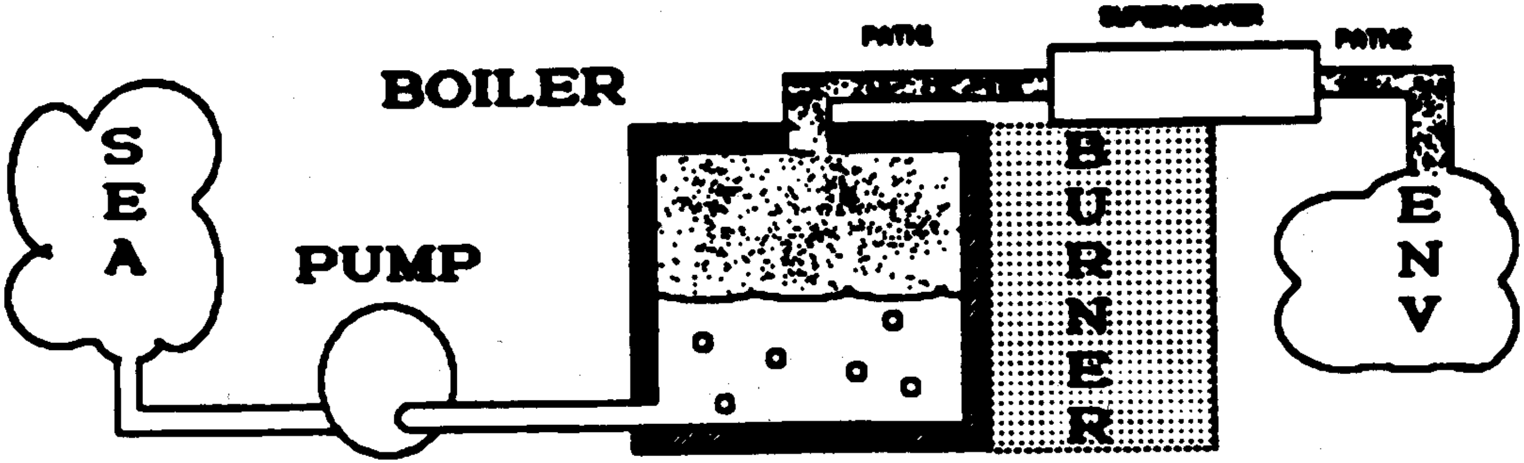
Two containers example

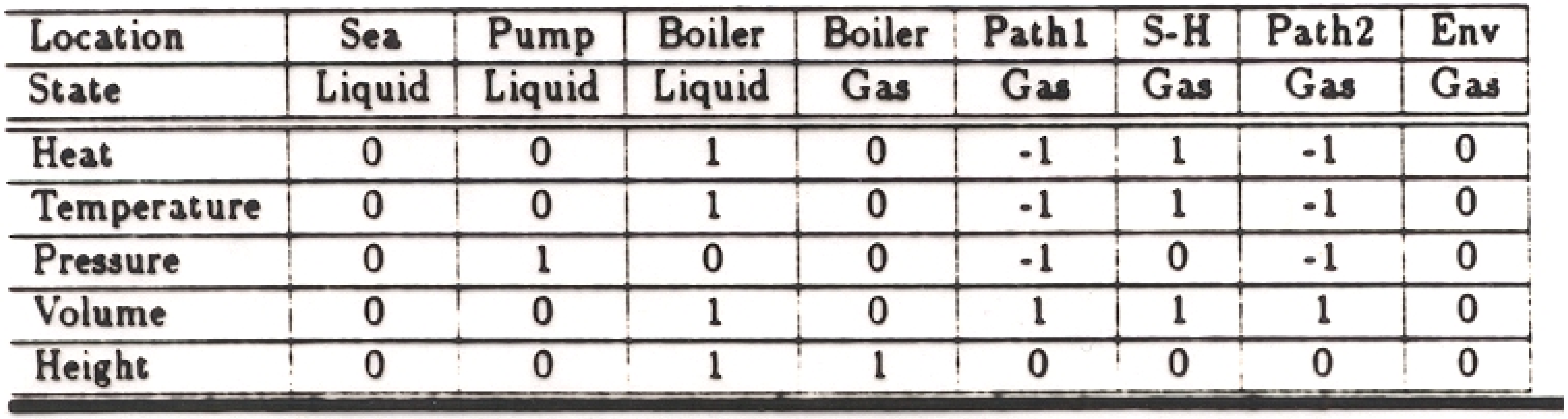




Location	Can1	Pump	Can2	F-P
Ds[Heat]	0	0	0	0
Ds[Temperature]	0	0	0	0
Ds[Pressure]	0	1	0	-1
Ds[Volume]	0	0	0	0
Ds[Height]	0	0	0	0

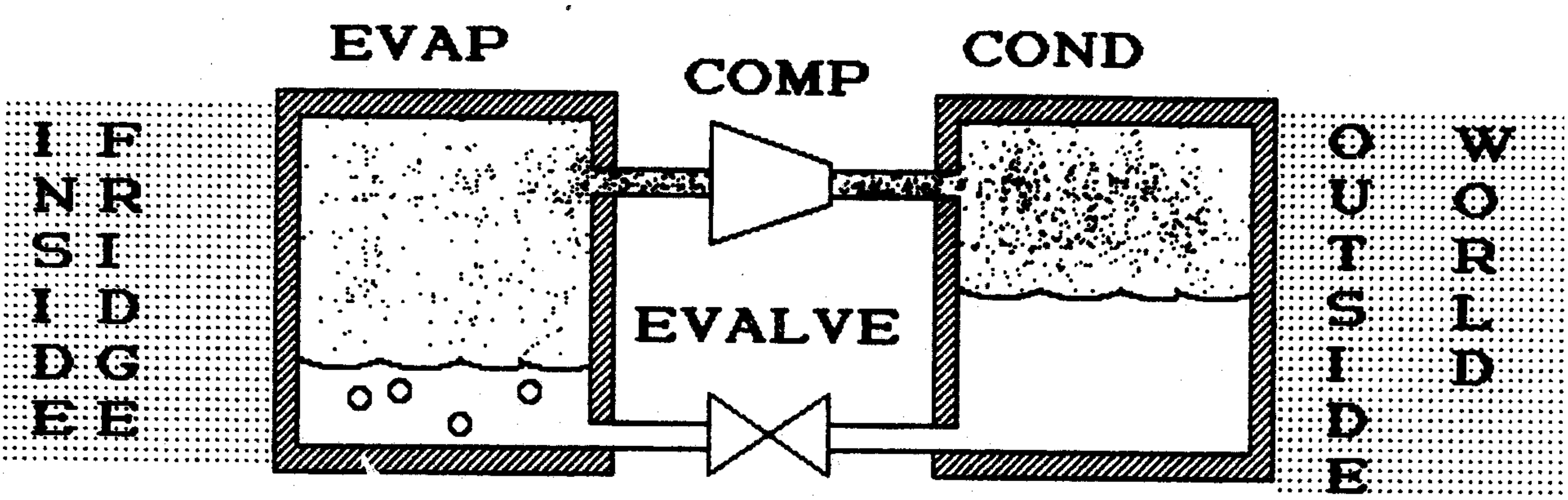
Steam plant example





Location	Sea	Pump	Boiler	Boiler	Path1	S-H	Path2	Env
State	Liquid	Liquid	Liquid	Gas	Gas	Gas	Gas	Gas
Heat	0	0	1	0	-1	1	-1	0
Temperature	0	0	1	0	-1	1	-1	0
Pressure	0	1	0	0	-1	0	-1	0
Volume	0	0	1	0	1	1	1	0
Height	0	0	1	1	0	0	0	0

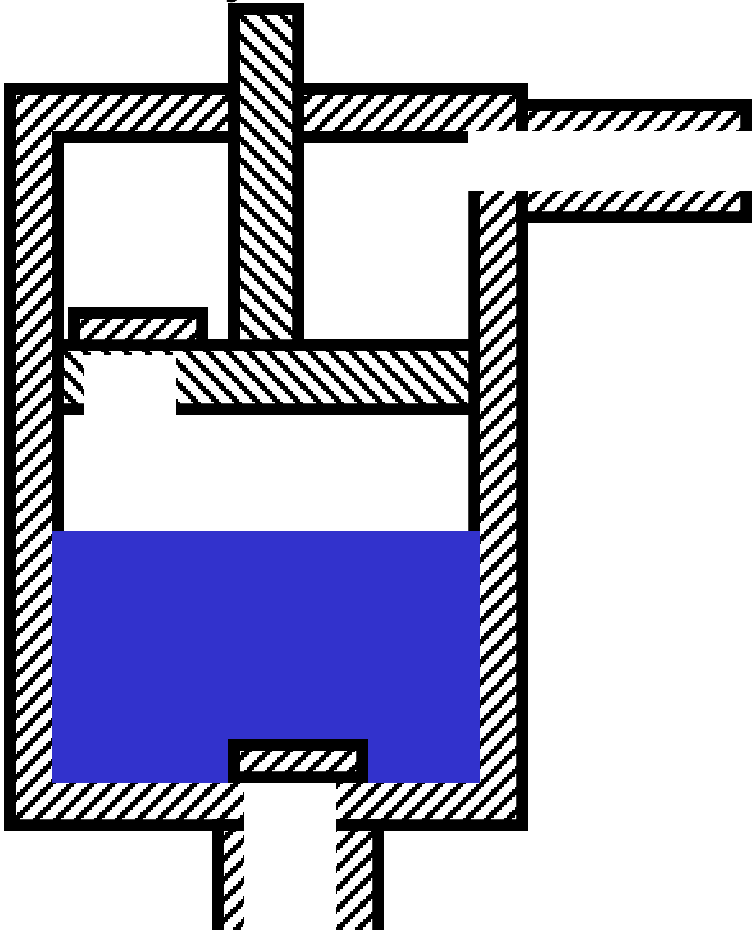
Refrigerator example



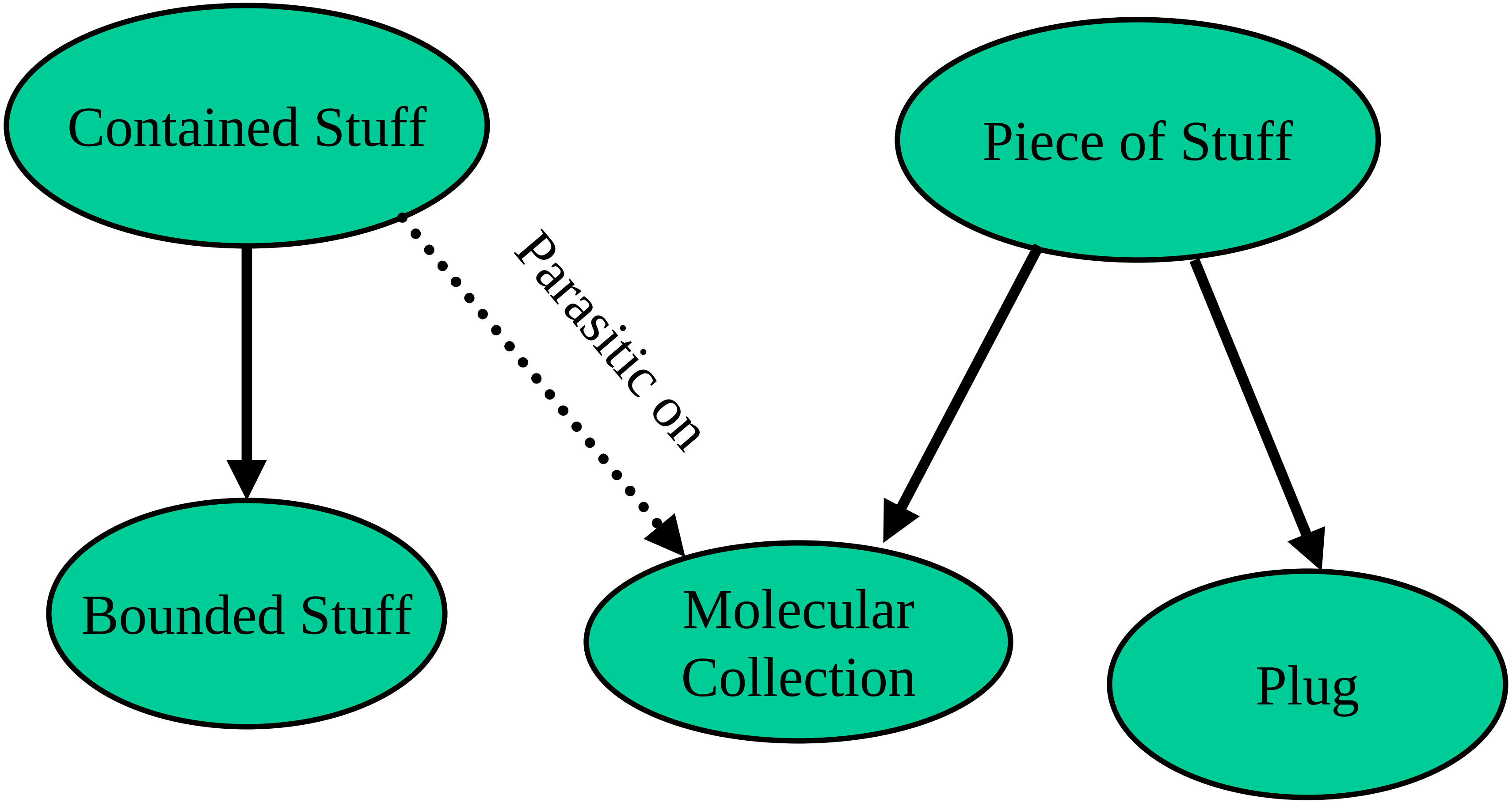
Location	Evap	Evap	Comp	Cond	Cond	EValve
State	Liquid	Gas	Gas	Gas	Liquid	Liquid
Ds[Heat]	1	0	1	-1	0	-1
Ds[Temperature]	-1	0	1	1	0	-1
Ds[Pressure]	0	0	1	0	0	-1
Ds[Volume]	1	0	-1	-1	0	0
Ds[Height]	1	1	0	-1	-1	0

Bounded stuffs

- Specialization of contained stuff ontology
- Where something is within the space matters
  - Affects connectivity



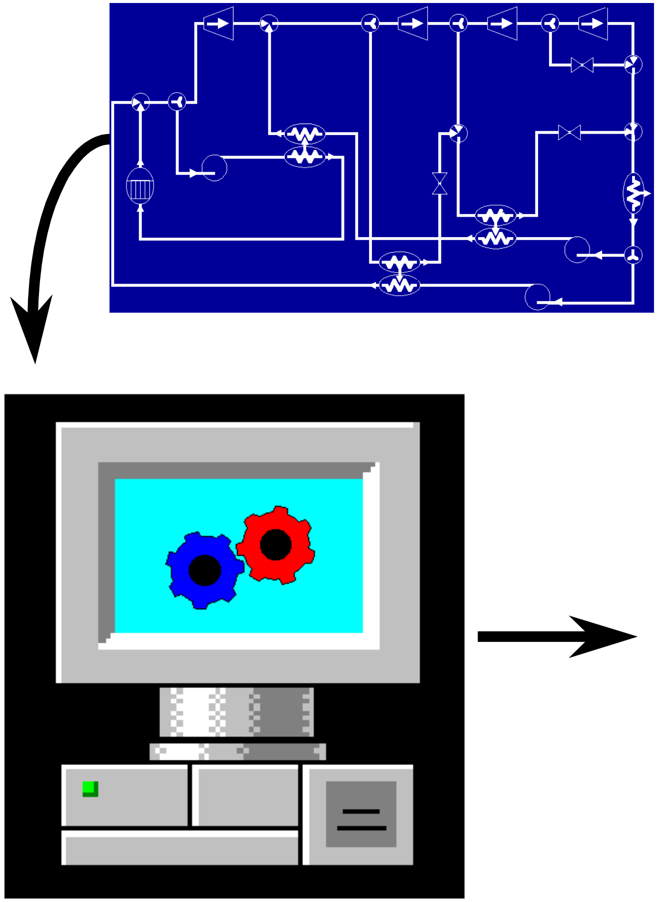
Ontology zoo for liquids



Function

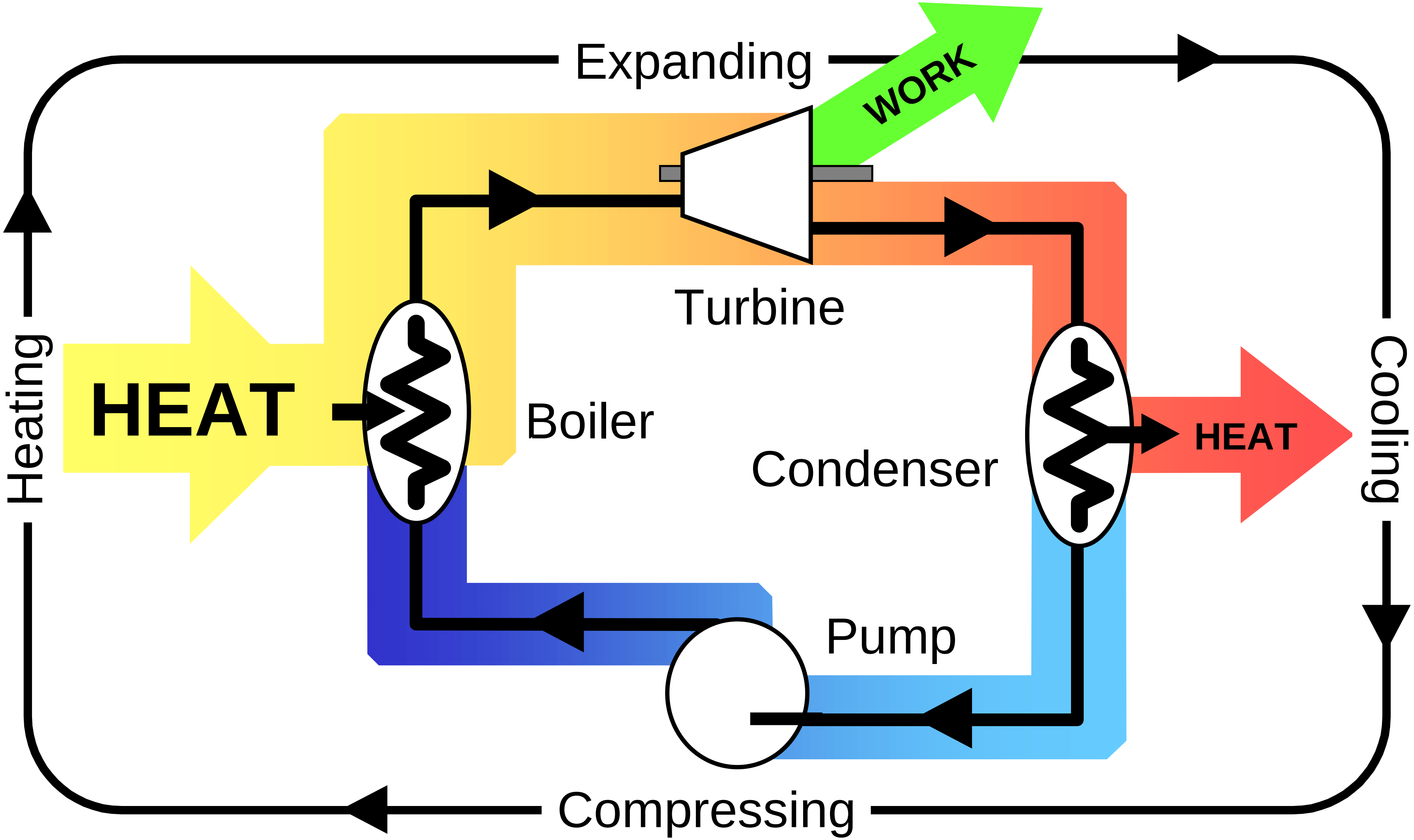
- Several approaches
  - Structure → Function, via qualitative simulation of behavior
    - One of the first tasks for QR, deKleer’s work in analog electronics
  - Structure → Function, via QR + evidential reasoning
    - Used in CyclePad, Everett’s work in engineering thermodynamics
  - Function as primary, used to generate behavior
    - Functional reasoning community
    - Insight: Often appropriate level for diagnosis, aspects of design

Goal: Automate Expert’s Teleological Inferences

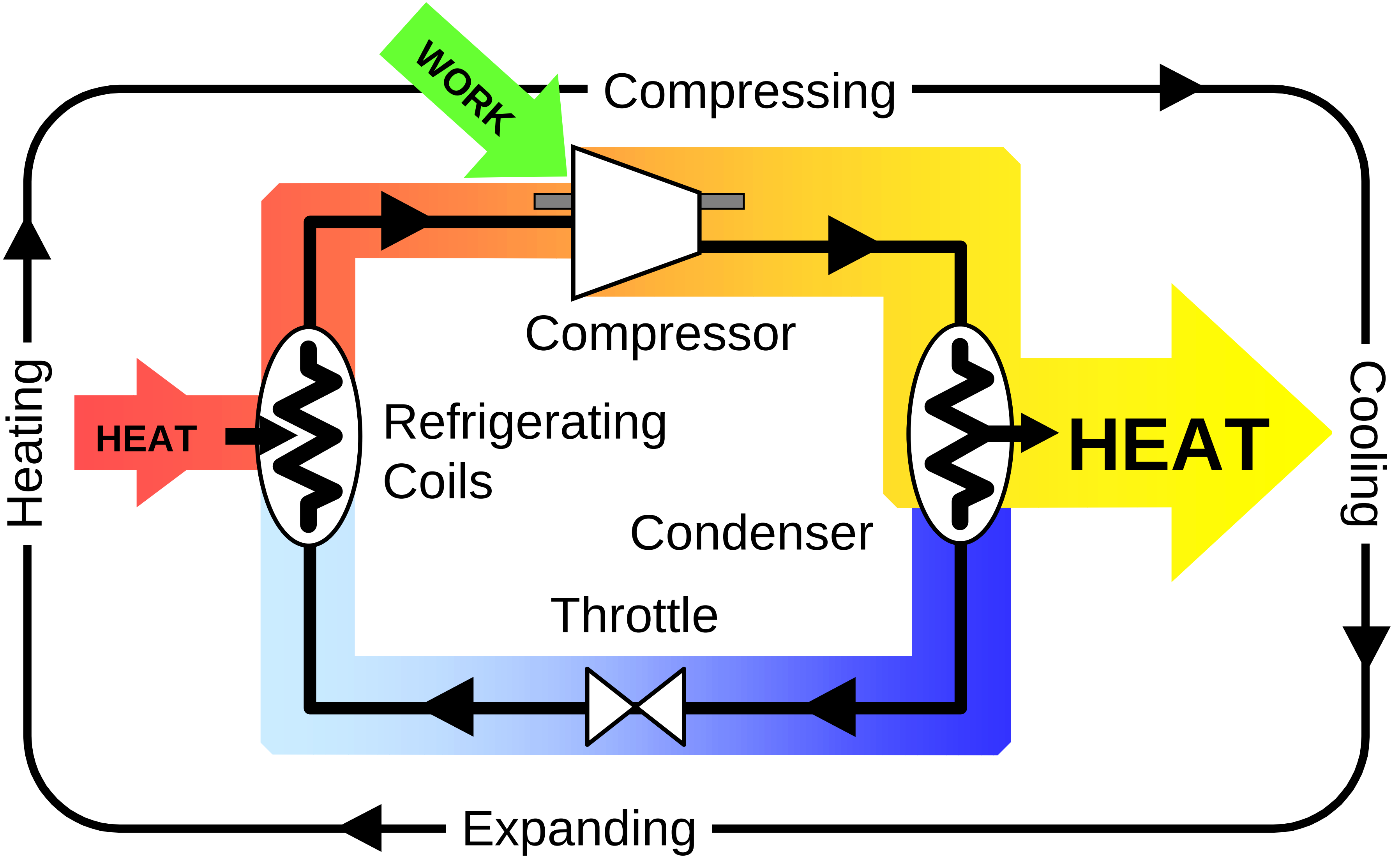


- Inference of student intent in a design-based intelligent learning environment
- Automatic indexing of schematics by function for retrieval by CAD and case-based systems
- Explanation of schematics to those using them

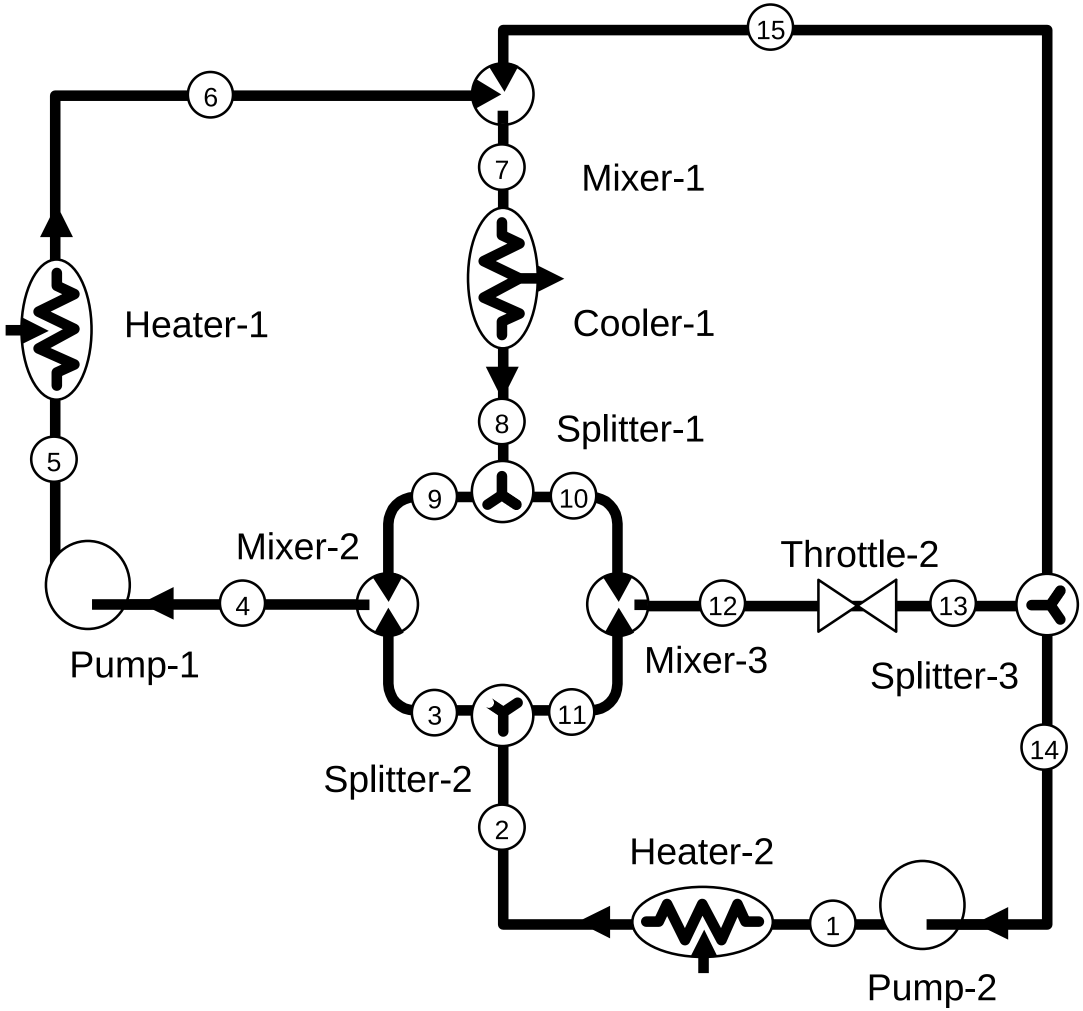
Heat Engines. . .



. . . and Refrigerators



Input: Schematic of a Jet-Ejection Air Conditioner

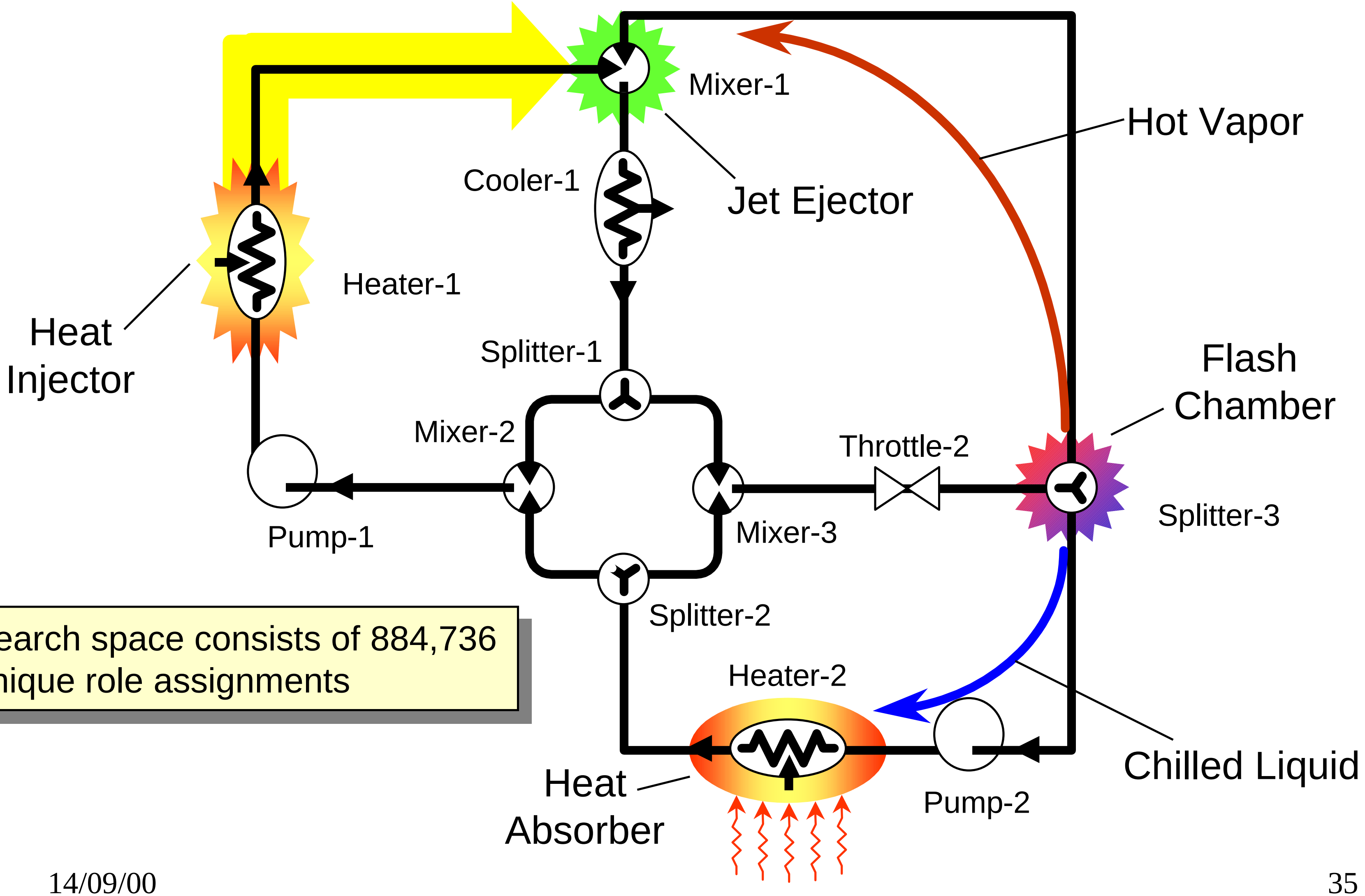


Representation of Input

(12 statements)

```
        :
(pump pmp1 s4 s5)
(heater htr1 s5 s6)
(mixer mxr1 s6 s15 s7)
        :
```

Roles of Components Depend on Structural Relationships



Search space consists of 884,736 unique role assignments

Output: Explanation of Device Roles. . .

- **What is the role of HTR-2?**
  `HTR-2 is acting as a HEAT-ABSORBER`

- **Why?**

  `HTR2 is construed as a HEAT-ABSORBER because refrigeration cycles are more likely to use heaters as heat-absorbers than as energy-injectors.`

- **Could HTR-2 be acting as a preheater?**

  `HTR2 is unlikely to be acting as a PREHEATER because a heater on a non-work-generating subcycle of a refrigerator is unlikely to be preheating the working fluid and a refrigerator rarely has need to preheat its working fluid.`

. . . System Teleology. . .

- **What is this system for?**

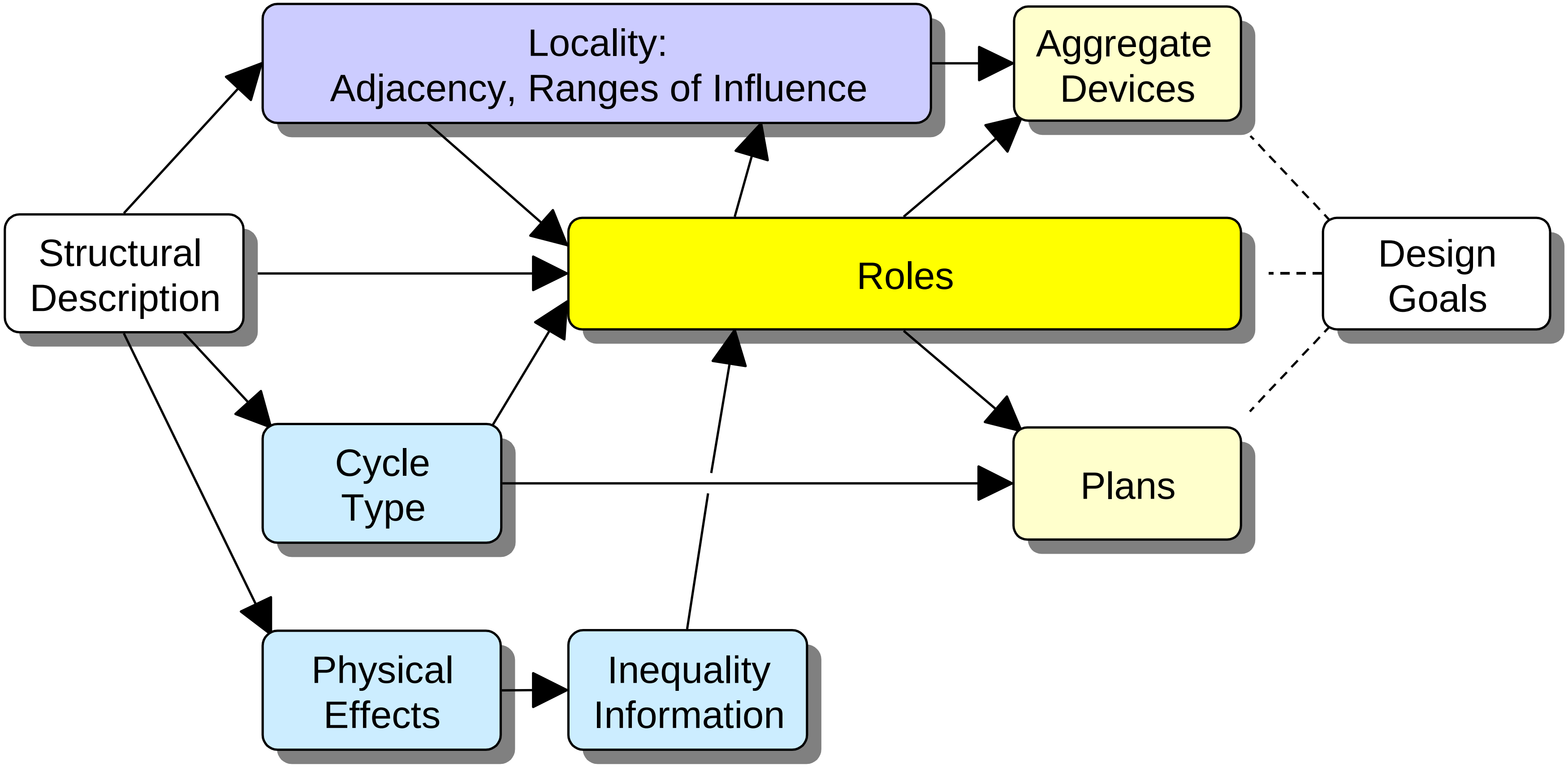
  This is a steam-jet-driven refrigeration system, with MXR-1 acting as a steam-jet compressor. The advantages of such a system are simplicity--no moving parts aside from feed pumps (PMP-1 and PMP-2), low cost, and safety, since such systems typically use water. However, because it cools via chilled liquid (in HTR-2), it cannot achieve low temperatures. Typical applications are for air conditioning, especially in passenger vehicles such as trains and ships.

. . . and System Behavior

- **Explain the function of this system**

  The liquid working fluid flowing through HTR2 absorbs heat from the environment. The heated working-fluid then flows to SPL2. SPL2 splits the working-fluid into two streams, one going to MXR2 and the other going to MXR3. MXR2 delivers working-fluid from SPL2 and SPL1 to PMP1. PMP1 delivers liquid working-fluid to HTR1. HTR1 vaporizes the working-fluid and delivers it to MXR1. MXR1 acts as a jet-ejection pump, powered by the stream of high-energy working fluid from HTR1. It compresses the vapor from SPL3 and delivers the resulting mixture to CLR1. CLR1 cools the working fluid. . .

Teleological Representations



Locality:
Adjacency, Ranges of Influence

Aggregate Devices

Structural Description

Roles

Design Goals

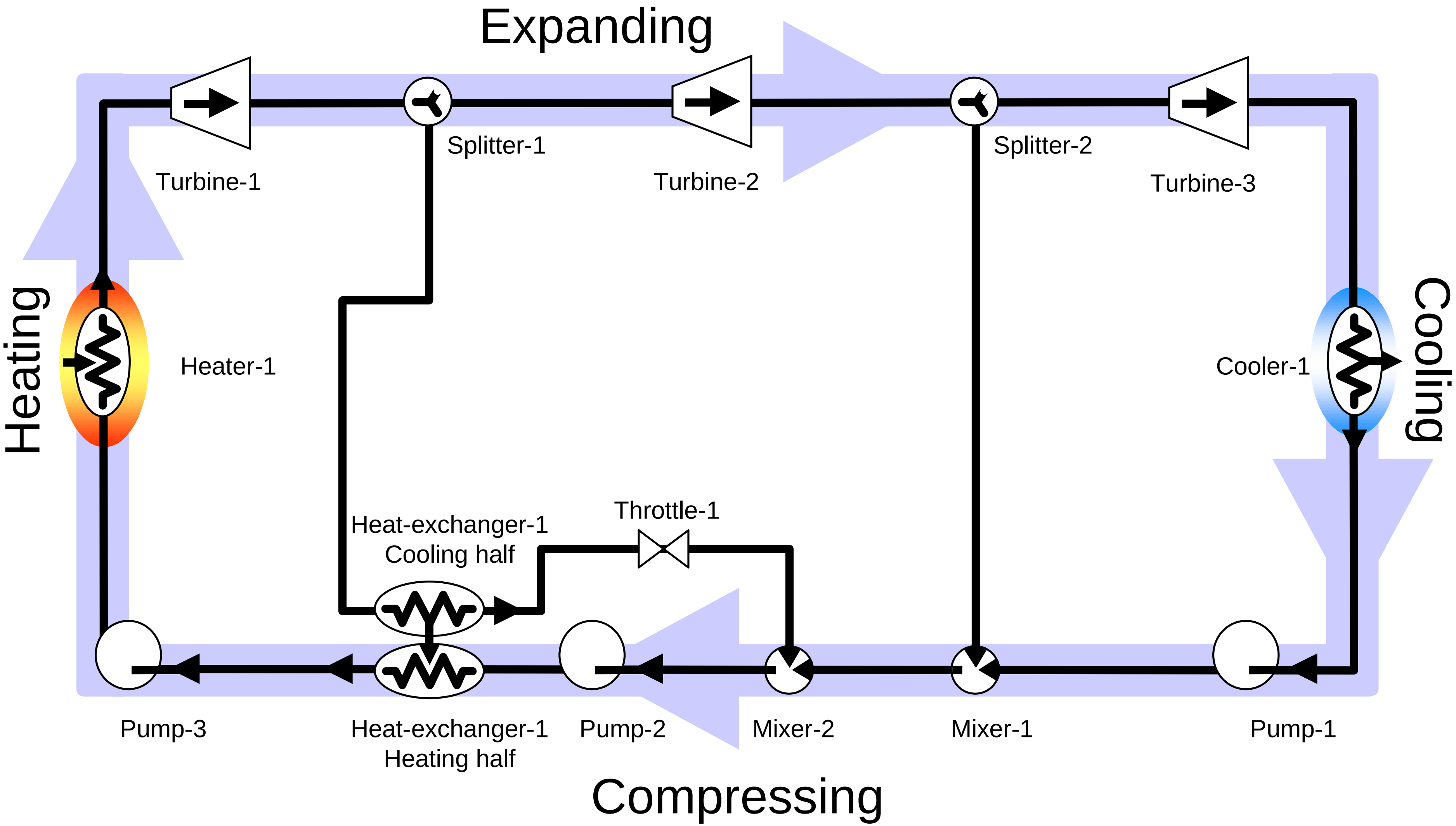
Cycle Type

Plans

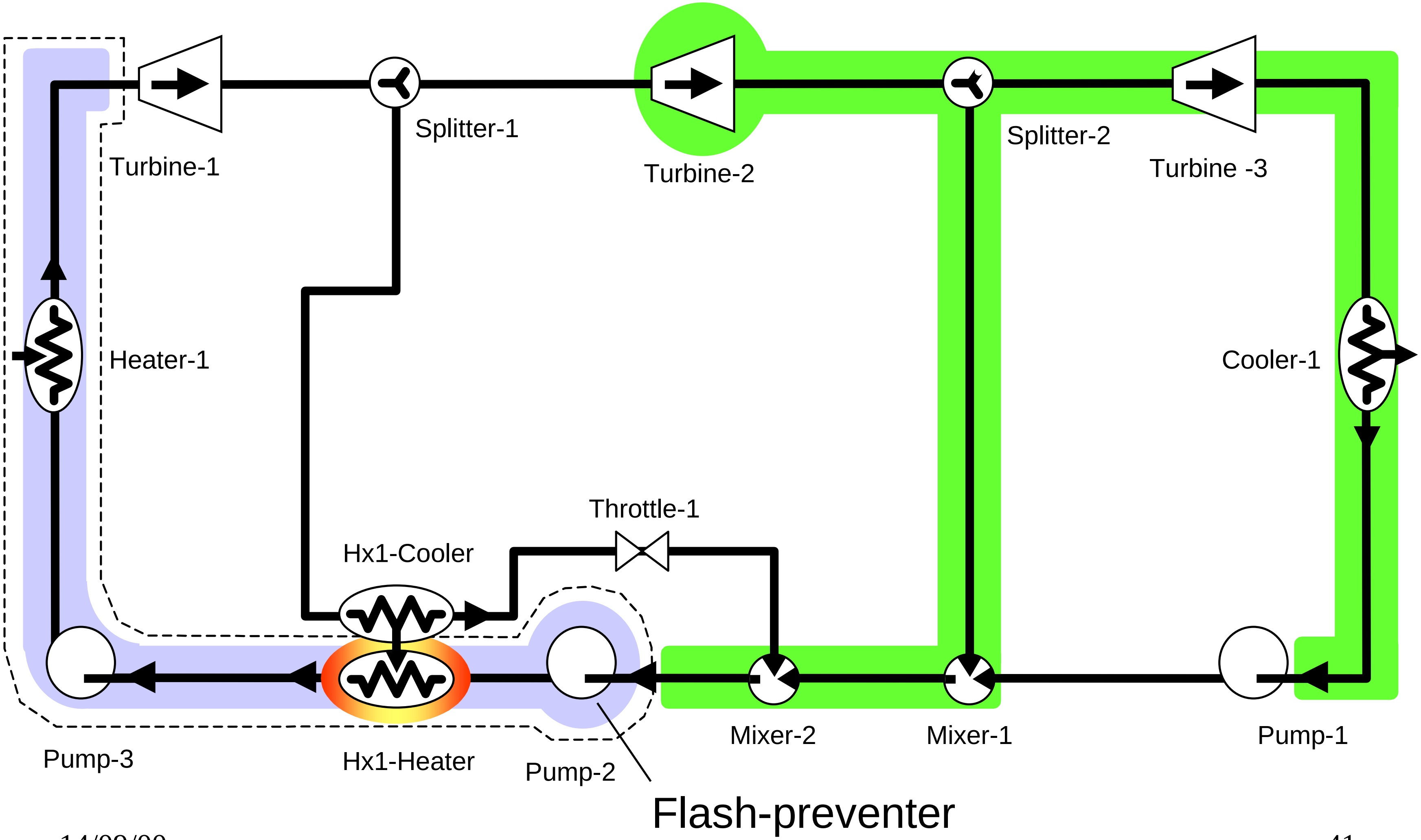
Physical Effects

Inequality Information

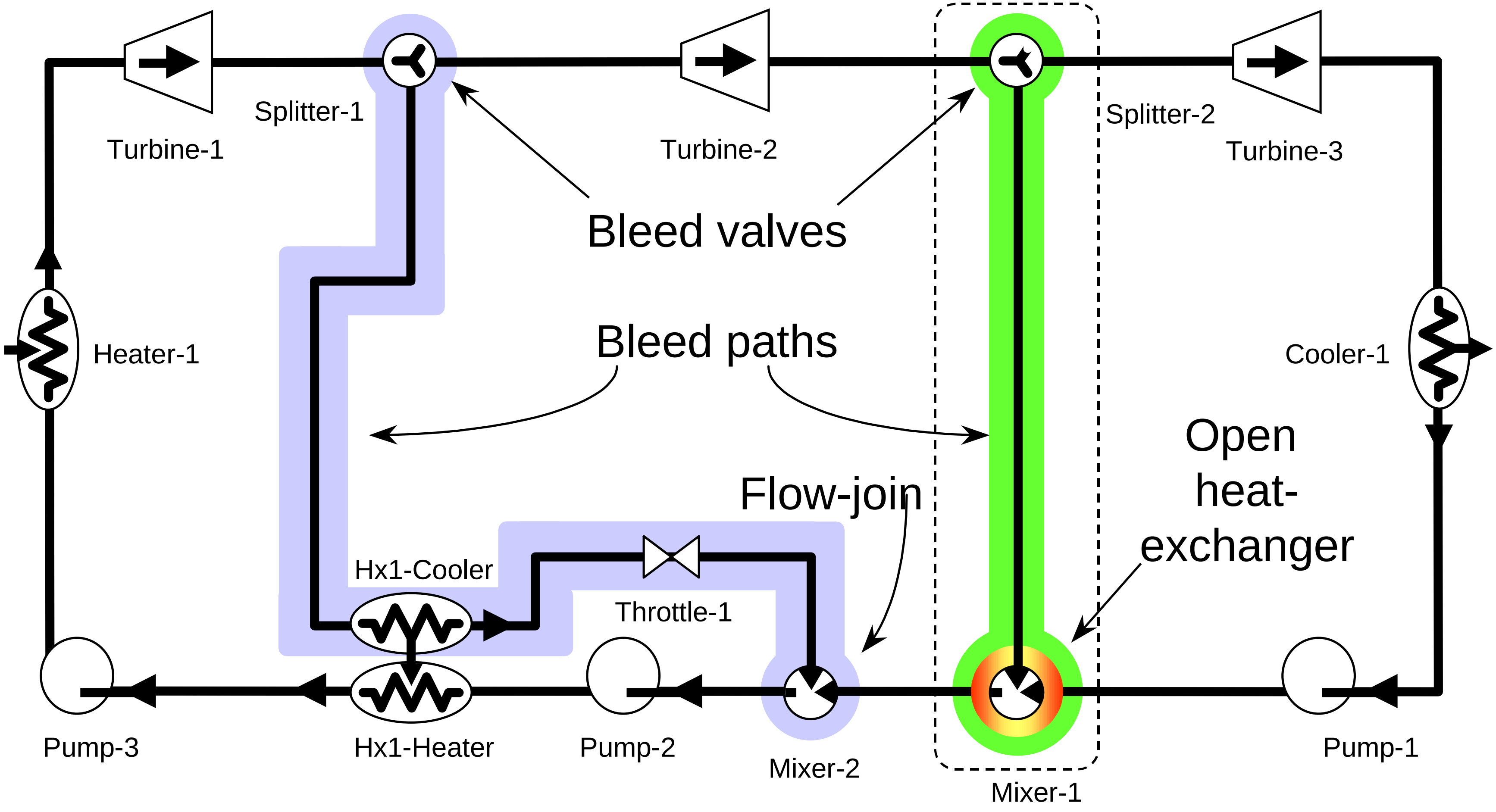
A Typical Power Plant: Rankine Cycle with Open and Closed Regeneration



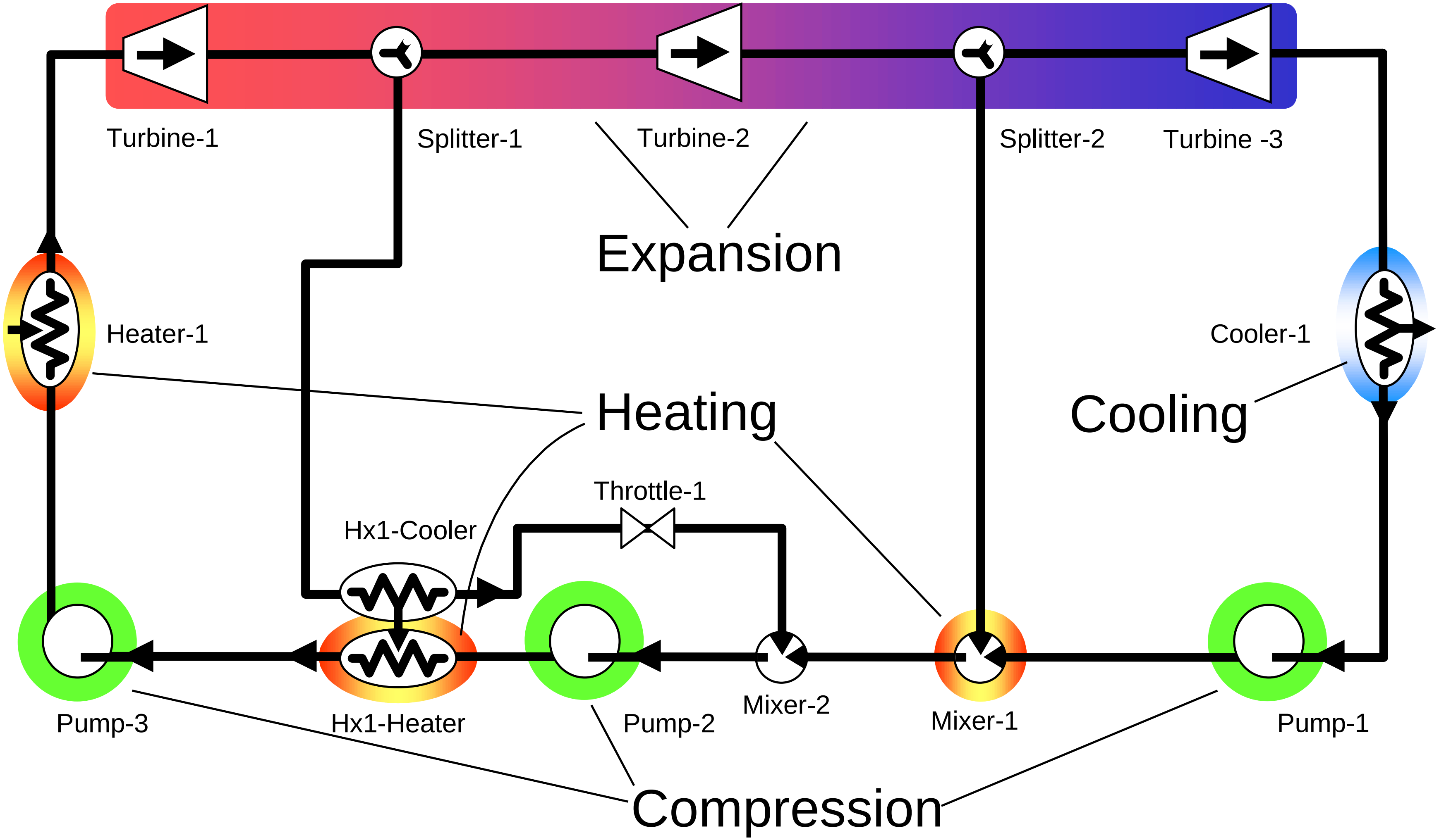
Ranges of Influence Provide More Flexibility in Definition of Locality



Recurring Teleological Patterns Help Describe Locality



Aggregate Devices
Provide Useful Abstraction



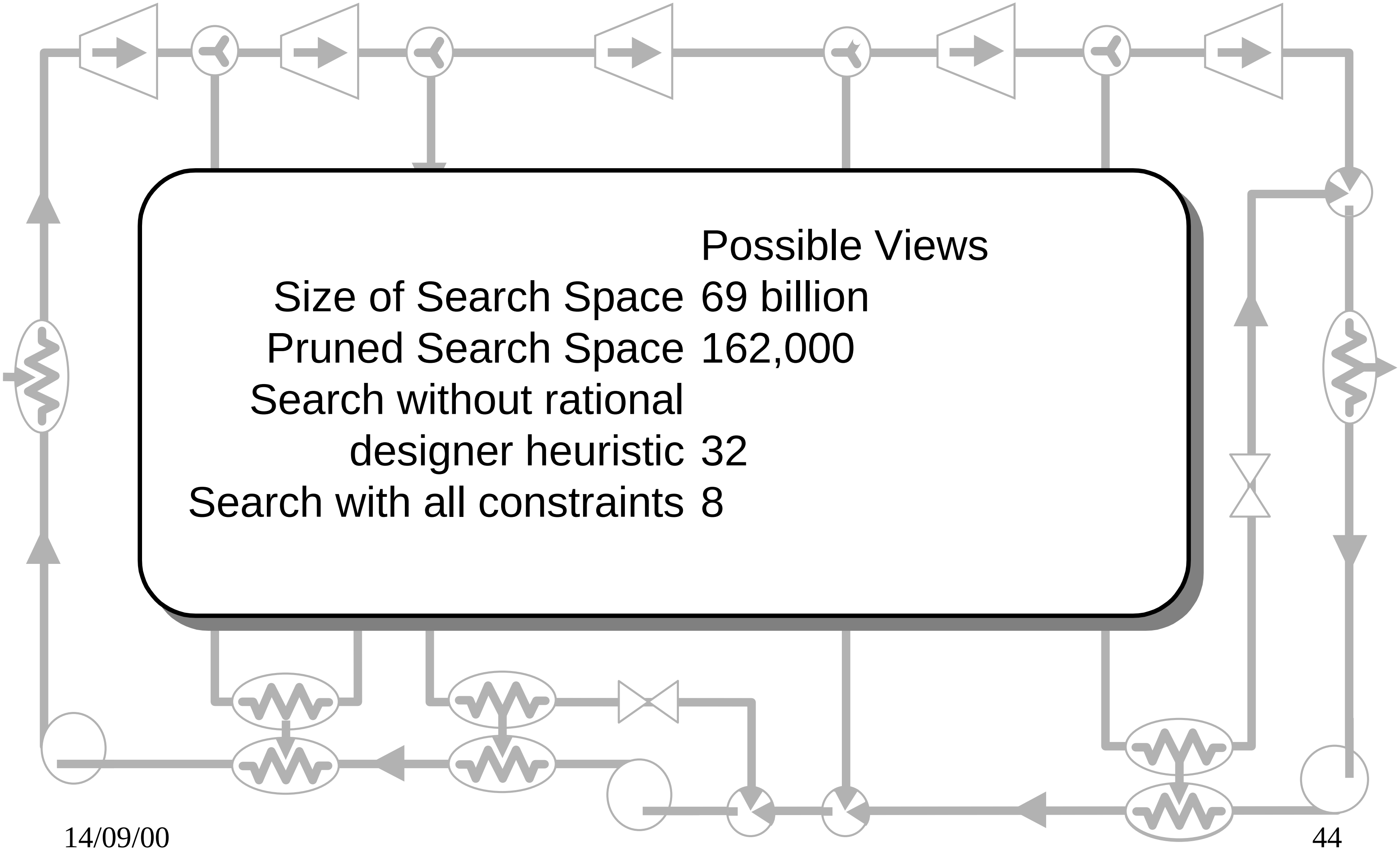
Expansion

Heating

Cooling

Compression

Ruling-in is Superior to Ruling-out



	Possible Views
Size of Search Space	69 billion
Pruned Search Space	162,000
Search without rational designer heuristic	32
Search with all constraints	8