Pyramid Problems in Soar & ACT-R

John Laird

26th Soar Workshop

Big Picture Goals

- Take instruction (not using NL)
  - Task instructions
  - Problem structure
- Execute task using domain-independent interpretation
  - No task-specific knowledge in rules
  - Except basic mathematics (7+6 = 13)
  - A few bits of special knowledge for meta-reasoning
- Match human data and compare to ACT-R
  - Chunking?
- Meta-Cognition

Mastering an Algebraic Concept

Pyramids:
There is a notation for writing repeated addition where each term added is one less than the previous:

For instance, 5 + 4 + 3 is written as 5 $ 2

Since 5 + 4 + 3 = 12 we would evaluate 5$2 as 12 and write 5$2 = 12

The parts of 5 $ 2 are given names:

5 is the base and reflects the number you start with
2 is the height and reflects the number of items you add to the base
5 $ 2 is called a pyramid

Instructions

Soar:

```
(<s1> ^action <a10> <a11> <a12>
      ^next <s2>)
(<a10> ^command set ^variable sum
       ^value 0 ^value-type constant)
(<a11> ^command set ^variable term
       ^value base ^value-type variable)
(<a12> ^command set ^variable count
       ^value 0 ^value-type constant)
(<s2> ^action <a30>
      ^next <s3>)
(<a30> ^command add ^variable sum
       ^value term ^value-type variable)
(<s3> ^action <a6>
      ^next <s4>)
(<a6> ^command goal-test ^relation equal
      ^variable count
      ^value height ^value-type variable
      ^type finished)
(<s4> ^action <a4> <a5>
      ^next <s2>)
(<a4> ^command decrement ^variable term)
(<a5> ^command increment ^variable count)
```

English

1. Set sum to 0

   Set term to base

   Set count to 0

2. Add term to sum

3. Test if count = height

4. Decrement Term

   Decrement Count

   Goto 2

Problem Structure and Example Problem

```
(<ps1> ^name base ^type variable ^next <ps2>)
(<ps2> ^name |$| ^type symbol ^next <ps3>)
(<ps3> ^name height ^type variable  ^next <ps4>)
(<ps4> ^name |=| ^type symbol ^next <ps5>)
(<ps5> ^name answer ^type variable ^next nil)
```

```
(<p1> ^value 5 ^type constant ^next <p2>)
(<p2> ^value |$| ^type symbol ^next <p3>)
(<p3> ^value 3 ^type constant ^next <p4>)
(<p4> ^value |=| ^type symbol ^next <p5>)
(<p5> ^value |?| ^type unknown ^next nil)
```

Basic Flow

- Initialize-instruction
- Initialize-problem
- Encode [Map problem onto problem structure]
  - Process-symbol, Process-variable, Process-unknown
- Execute-solve-procedure [Interpret procedure to solve problem]
  - Execute-steps
    - Set, Add, Subtract, Increment, Decrement, Goal-test
  - Next-step
- Write-answer [Write out the answer]
- Reflect - [Looks for patterns in problems]
  - Detect first-term - height = last-term
  - Detect balanced problems around 0
- Next-problem

Evaluation Problems

1. 5 $ 3
   5 + 4 + 3 + 2 = 14
2. 10 $ 4
   10 + 9 + 8 + 7 + 6 = 40
3. 8 $ 1
   8 + 7 = 15
4. 3 $ 4
   3 + 2 + 1 + 0 + -1 = 5
5. 5 $ 7
   5 + 4 + 3 + 2 + 1 + 0 + -1 + -2 = 12
6. 0 $ 4
   0 + -1 + -2 + -3 + -4 = -10
7. 13 $ 0
   13
8. 1000 $ 2000
   $\underbrace{1000 + \ldots + 1 + 0 + -1 + \ldots + -1000}_{2000} = 0$

Expression Writing Problems

9. $6 + 5 + 4 + 3$
   6$3
10. $9 + 8 + 7$
    9$2
11. $1 + 0 + (-1) + (-2)$
    1$3
12. $x + (x - 1) + (x - 2) + (x - 3) + (x - 4)$
    x$4
13. $20 + (20 - 1) + \ldots + (20 - 11)$
    20$11
14. $15 + (15 - 1) + \ldots + (15 - x)$
    15$x
15. $z + (z - 1) + \ldots + (z - y)$
    z$y

Find the Height Problems

16. 6 $ x =15
    6 + 5 + 4 = 15 --> x = 2
17. 10 $ x =55
    10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 = 55 --> x = 10
18. 912 $ x =912
    x = 0
19. 3 $ x = -9
    3 + 2 + 1 + 0 + -1 + -2 + -3 + -4 + -5 = -9 --> x = 8
20. 100 $ x = -101
    100 + … + 1 + 0 + -1 + …-100 + -101 = -101 --> x = 201

201

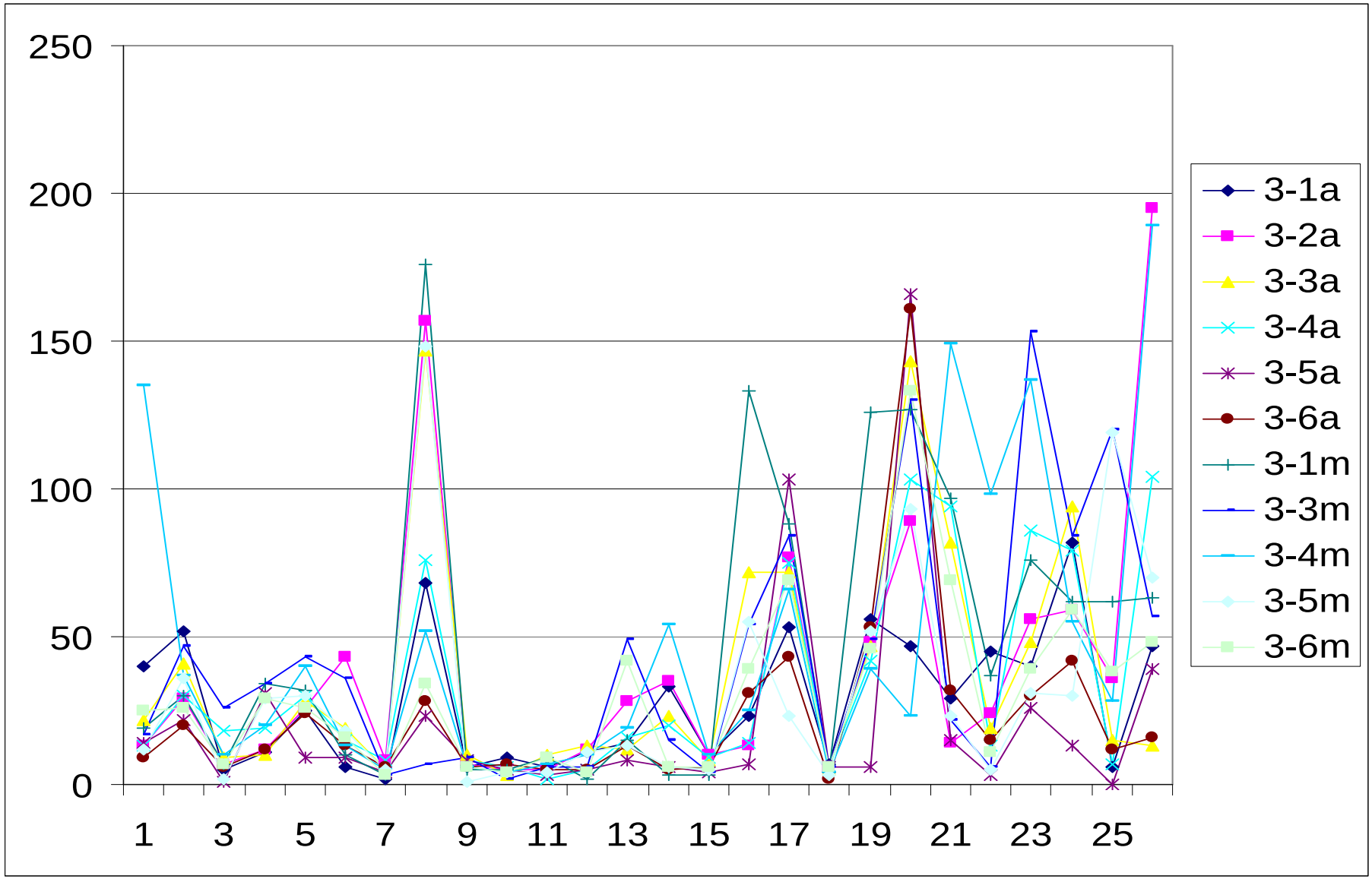
Find the Base Problems

21. x $ 2 =15
    guess and check: 7+6+5 = 18; 6+5+4 = 15 or
    x + (x - 1) + (x -2) = 15 --> 3x - 3 = 15 --> x = 6
22. x $ 1 = 15
    x = 8
23. x $ 4 = 35
    x = 9
24. x $ 6 = 35
    x = 8
25. x $ 6 = 0
    x = 3
26. x $ 6 = -7
    x = 2

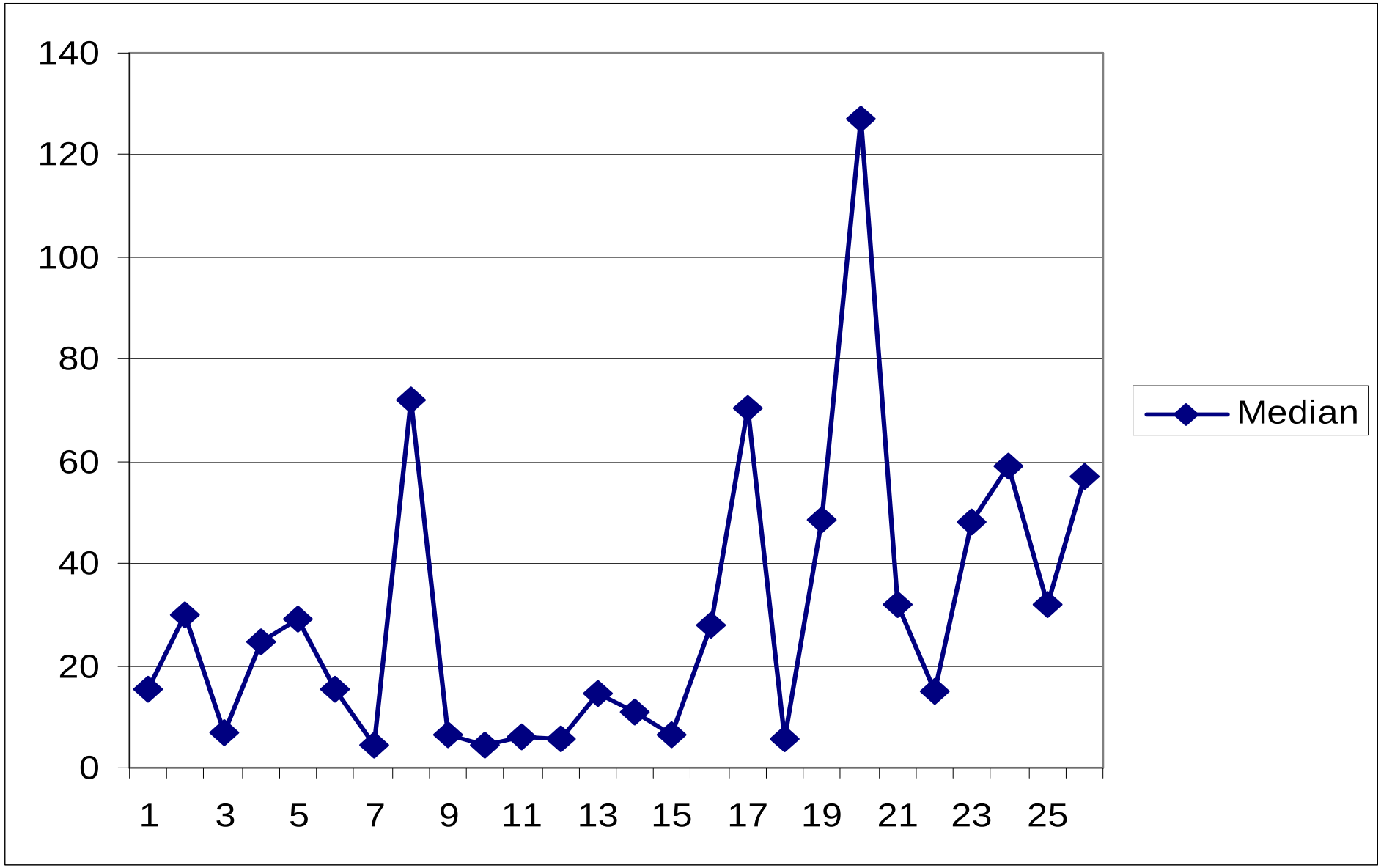
Soar Approach to Problem Types

- Solve: 5$3 =
  - Uses execution procedure
- Describe: 6+5+4
  - Uses describe procedure (what ACT-R does too)
- Solve: 6$x=15
  - Uses execution procedure - stops when answer achieved: Learned stop by doing first set of problems
- Solve: X$2=15
  - Impasses on setting Base = X
    - Generate and tests values of X and then solves
      - Must create hypothetical problems
    - If fails, then must generate a new guess
    - Smart generator (based on prior problem, prior guesses)

Individual Human Data



Median Human Data

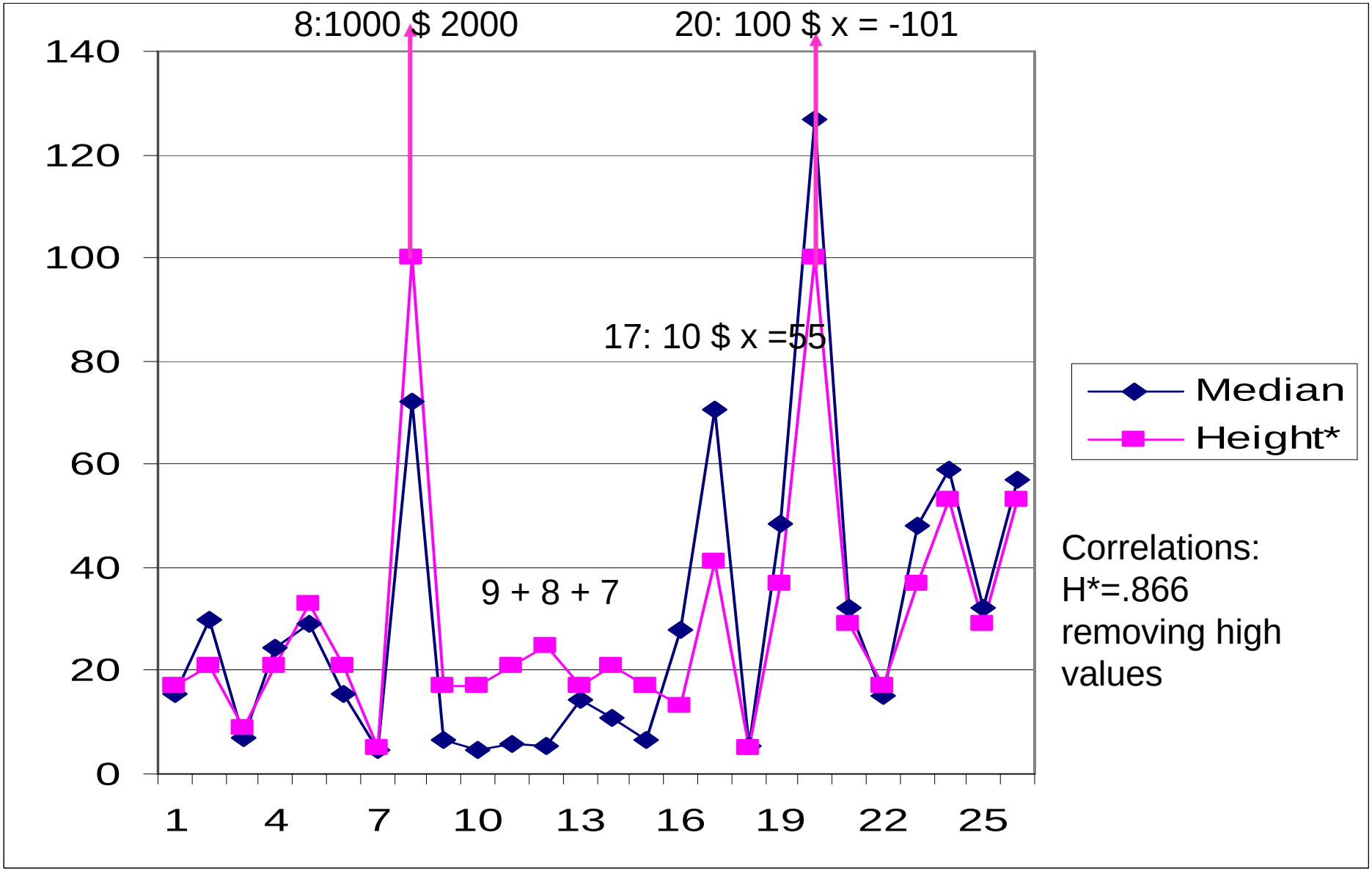


Simple Model: Height*

- Time is proportional to Height
  - Base $ Height = ?
  - This is clearly the most important part of the procedure

- Extend to take into account finding base problem
  - X $ 2 = 15
  - Simple model of guessing X, modifying guess if wrong.

Median, Height*



Comments on 1000$2000: John Anderson

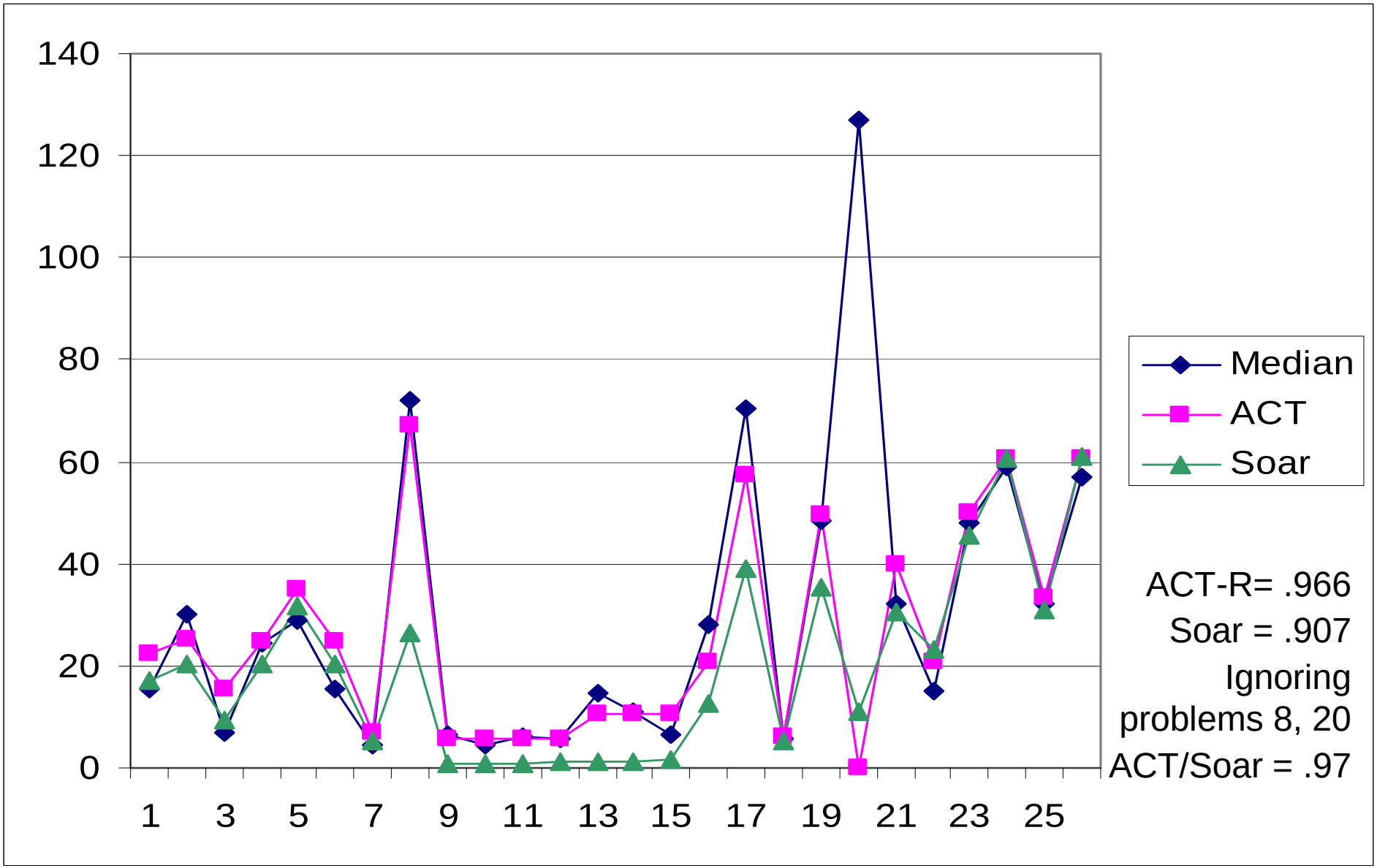
1. Students averaged about half of their time in unproductive attempts before they tried a method that work.
2. An unproductive path tried by many was to find an analogy to what they knew about factorial.
3. Five students reasoned about simpler problems like 2$4.
4. Others reasoned more abstractly.
5. A number of students confirmed the answer (0) by a second method before giving it as their final answer.
6. The final ACT-R model tried factorial, then abstract reasoning, and finally confirmed by solving 2$4.
7. Two significant issues for modeling are interrupting regular processing and accumulating needed knowledge.
8. Both are metacognitive in that they require parallel reflection on the ongoing problem solving

Soar Approach to 1000$2000

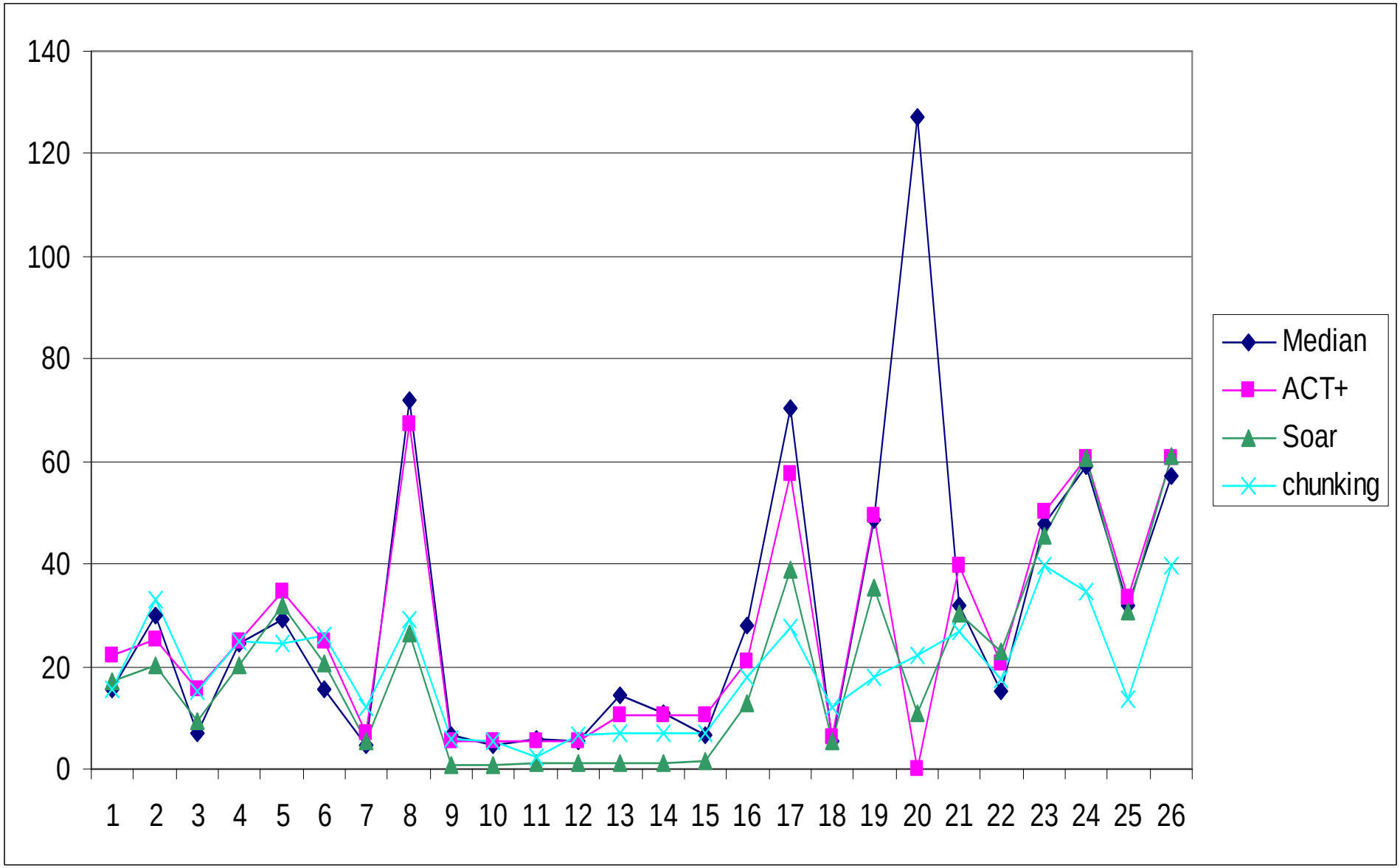
- *Detects "large" height*
- Attempts "abstract" solution
  - What can it compute?
  - First-term: 1000, Last-term: -1000 (derived from observed relation)
  - *Notice "balanced": 1000, -1000 => 0*
- Create simple problem to check
  - *Creates 2$4=*
  - Solve simple problem => 0
- Assumes that is the answer

- Special prior knowledge:
  - *Detect large height*
  - *Note balanced*
  - *Simple problem generator*
- Soar doesn't mess around with factorial, etc. like ACT model and humans do but clearly could.

Median, ACT, Soar (scaled)

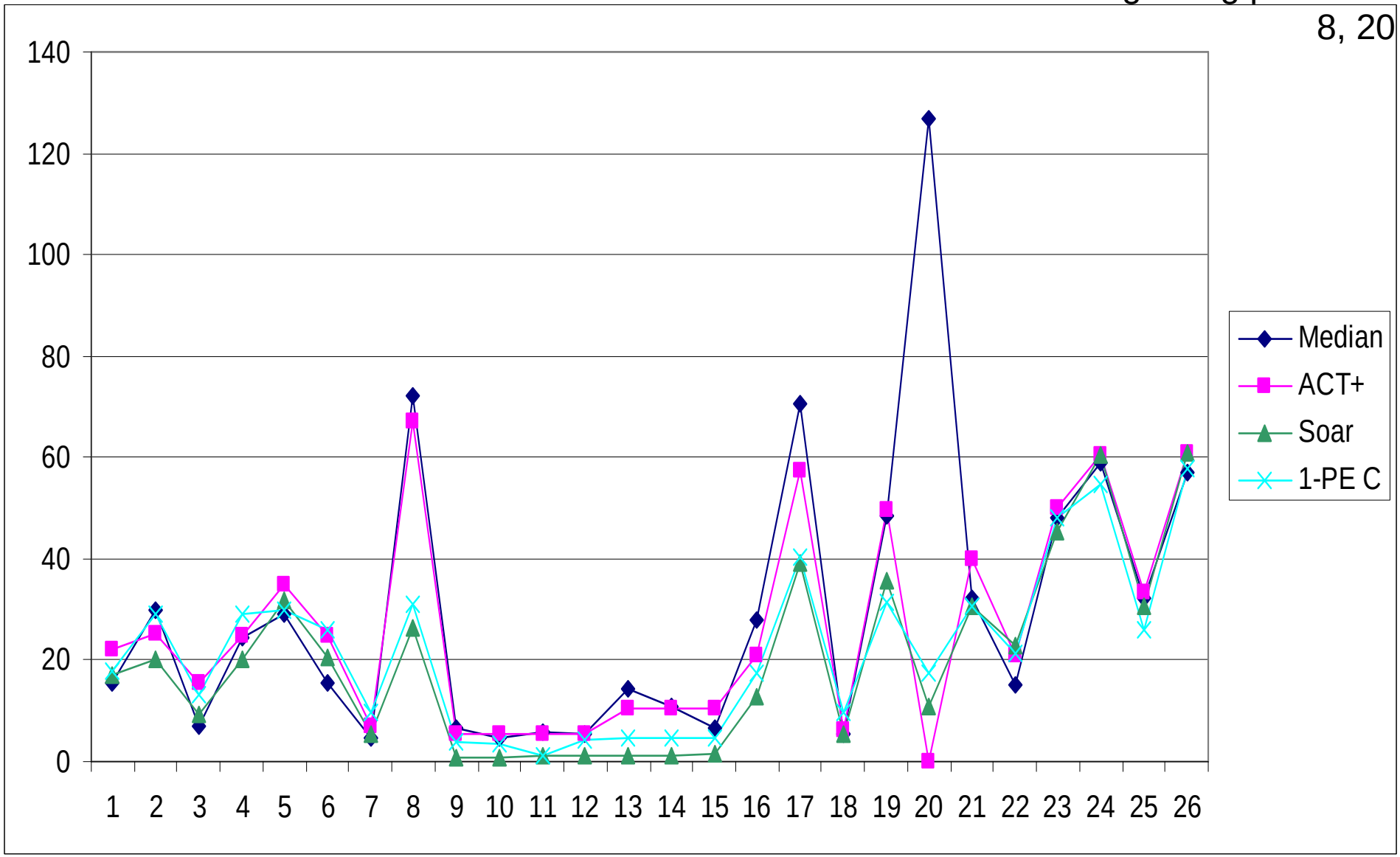


Median, ACT+, Soar, Soar w/ Chunking

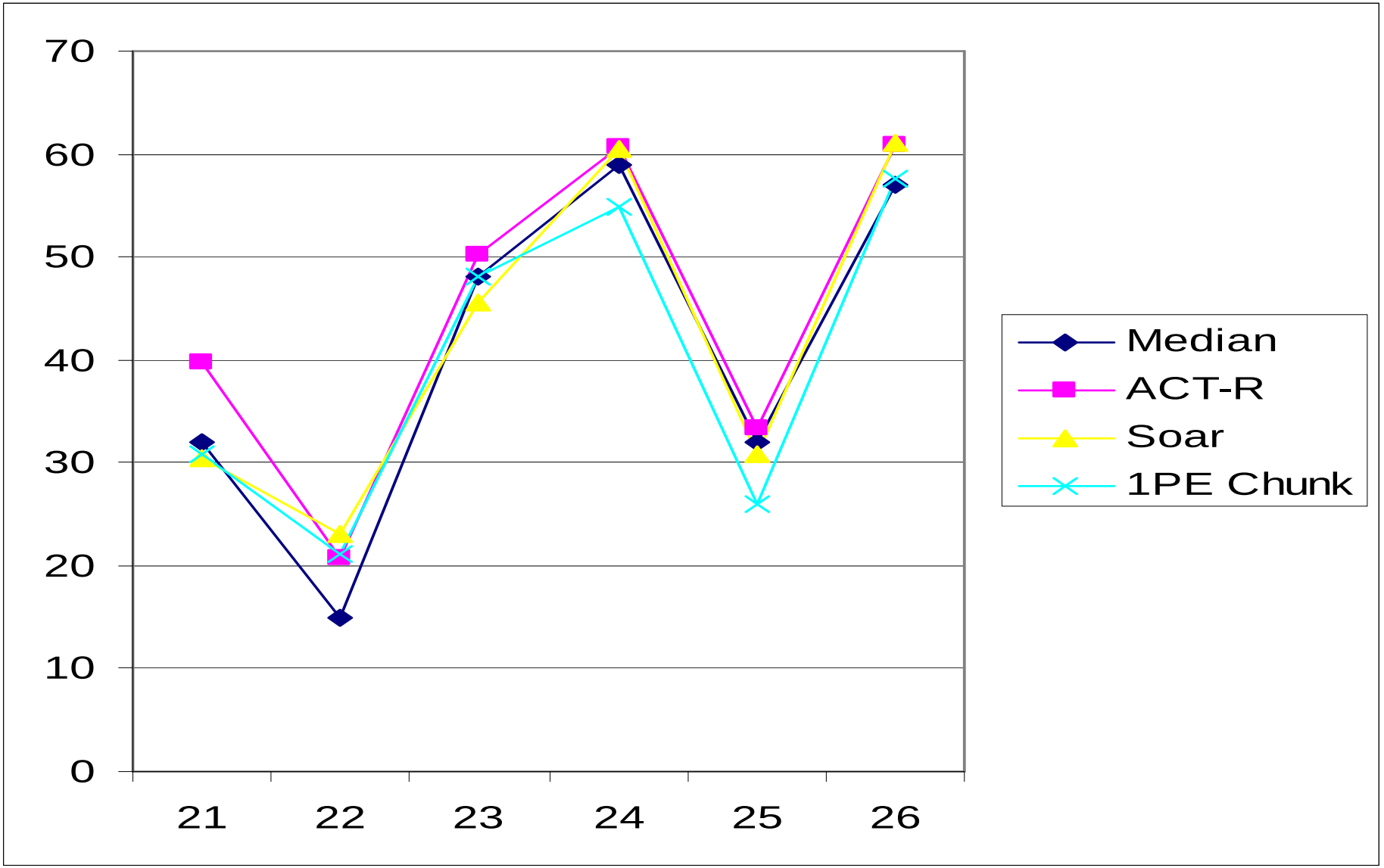


Chunking with 1PE/Decision

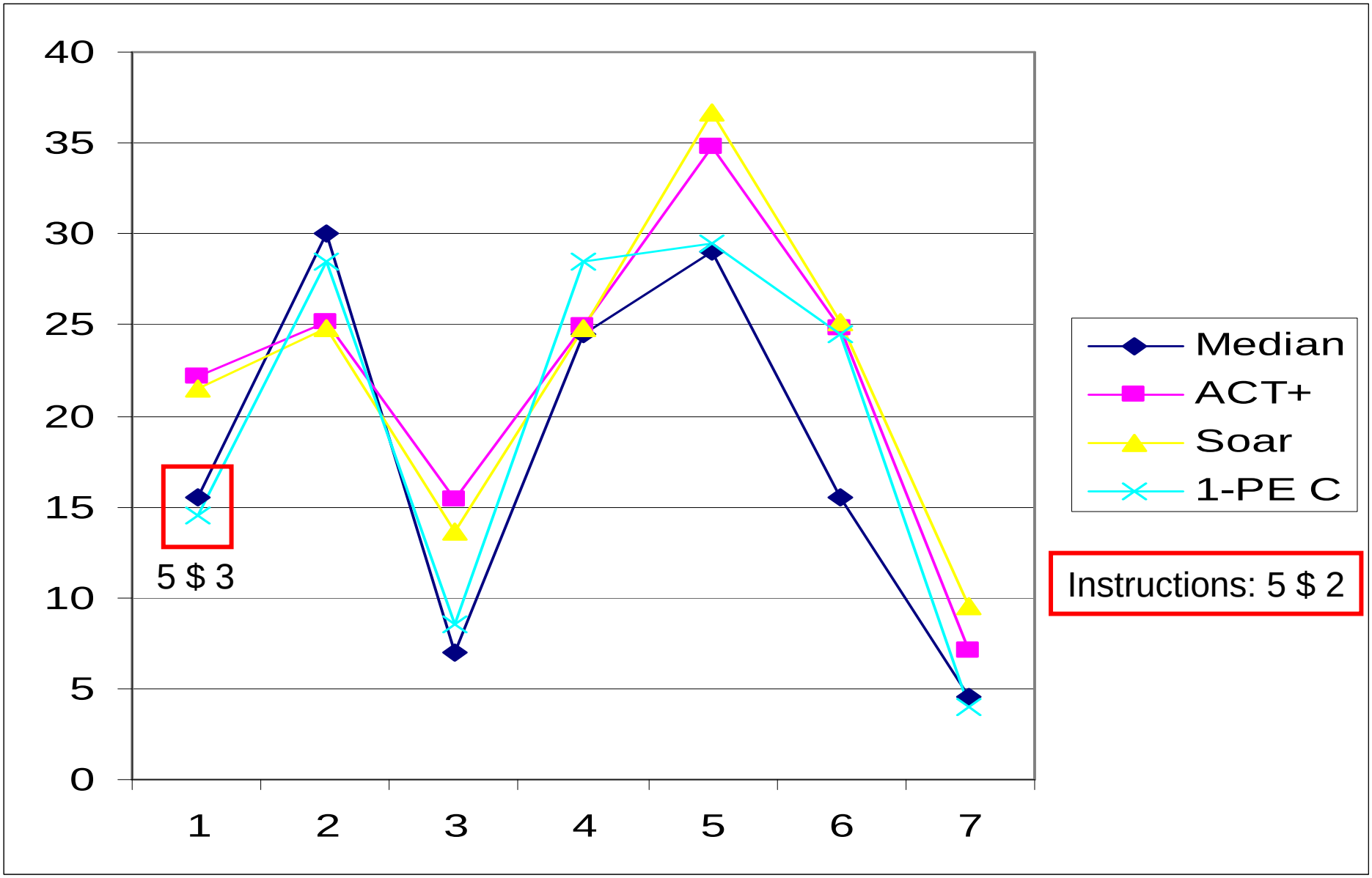
ACT-R= .966
Soar = .907
1PE = .906
Ignoring problems
8, 20



Last 6 problems



First 7 problems



Conclusions

- Nuggets:
  - Can do instruction taking (again)
  - Leads to surprisingly good results
    - It is (almost) all about doing the task (following instructions)
    - Results hold up with chunking 1PE/Decision
  - Soar is natural for metacognition
    - Impasses
    - Creating test problems in subgoals
    - Reasoning about structures complex structures (variable attributes)
- Coal:
  - More work to do on detailed comparison with ACT-R
  - More work on where some extra knowledge comes from
  - Soar model is scaled
    - Not 50msec/decision
    - No model of perception, …