Two NP-complete problems useful for reducing to arithmetic (summing) problems:

\(2\)-Partition: given integers \(A = \{a_1, a_2, \ldots, a_n\}\), partition \(A\) into two sets \(A = A_1 \cup A_2\) of equal sum: \(\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} a_i = t\)  
[\text{Karp 1972}]

**Generalization:** Subset Sum
\[ t = \sum_{i=1}^{n} \frac{a_i}{2}\]
given integers \(A = \{a_1, a_2, \ldots, a_n\}\), and a target integer \(t\), find a subset \(S \subseteq A\) of sum \(\sum_{i \in S} s = t\)

3-Partition: given integers \(A = \{a_1, a_2, \ldots, a_n\}\), partition \(A\) into \(\lceil n/3 \rceil\) sets \(A_i\) of equal sum, \(\sum_{i=1}^{\lceil n/3 \rceil} A_i = t\)

- can assume each \(a_i \in (t/4, t/2)\)
- each set \(A_i\) contains exactly 3 items
  [\text{Garey & Johnson - SICOMP 1975}]

\(\Rightarrow\) can make each \(a_i\) close to \(t/3\): add huge number \((n^{100} \cdot \text{max } A)\) to each \(a_i\)

Garey & Johnson [book] reduce
3SAT \(\rightarrow\) 3DM \(\rightarrow\) 4-partition \(\rightarrow\) 3-partition \(\rightarrow\) numerical 3DM
Variation: Numerical 3-dimensional matching

given integers \( A = \{a_1, a_2, \ldots, a_n \} \),
\( B = \{b_1, b_2, \ldots, b_n \} \),
\( C = \{c_1, c_2, \ldots, c_n \} \)

partition into \( n \) triples \( S_i \subset A \times B \times C \)
of equal sum \( t = \sum (A_i \cup B_i \cup C_i) / n \)

[Garey & Johnson - SICOMP 1975]

Reduction to 3-partition: (So it's simpler)
- add \( \varepsilon \ll 1 \) to each \( a_i \); e.g. \( \varepsilon = \frac{1}{4} \)
- add \( S \ll \varepsilon \) to each \( b_i \); e.g. \( S = \frac{1}{16} \)
- subtract \( \varepsilon + S \) from each \( c_i \)
- scale back to integers \( \times 16 \)
- in sum of 3, \( S \) never becomes \( \varepsilon \)
& \( \varepsilon \) never becomes 1
\( \implies \varepsilon \) & \( S \)s must cancel algebraically
\( \implies \) cf. (2D) matching

Generalization: 3-dimensional matching (3DM)
given a tripartite hypergraph with
vertices \( A \cup B \cup C \), \( |A| = |B| = |C| = n \),
& hyperedges \( E \subseteq A \times B \times C \),
find \( n \) disjoint edges \( S \subseteq E \)
(which must partition the vertices)

[Karp 1972]

Generalization: Exact Cover by 3-sets (X3C)
given 3-uniform hypergraph \((V, E)\),
\( \forall e \in E: 1 \leq l(e) = 3 \)
find \( |V| / 3 \) disjoint edges \( \implies \) partition \( V \)
*Two types of NP-hardness for number problems:

Weakly NP-hard = NP-hard
- allow numbers to have value exponential in n
- encoding length = \( \log(2^n c) = n^c \) still polynomial

Strongly NP-hard = NP-hard even when restricted to numbers with value polynomial in n
(i.e. even if numbers encoded in unary)

*Corresponding algorithmic notions:

Pseudopolynomial = polynomial in n & largest number
Weakly polynomial = polynomial = polynomial in n & \( \log(\text{largest number}) \)
Strongly polynomial = polynomial in n

Weak NP-hardness precludes polynomial algorithm (assuming \( P \neq NP \)) but leaves possible pseudopolynomial

Assuming \( P \neq NP \):  
\[ \text{strongly polynomial} \rightarrow \text{weakly NP-hard} \rightarrow \text{strongly NP-hard} \rightarrow \text{difficulty} \]
Multiprocessor scheduling: [Garey & Johnson - SICOMP 1975]
- given $n$ jobs with processing times $a_1, a_2, \ldots, a_n$
- given $p$ processors (each sequential & identical)
- assign jobs to processors to minimize maximum completion time (makespan)
- decision version: can all processors finish by $\leq t$?
- $\textbf{NP certificate: job } \mapsto \text{processor mapping}$
  $\quad (a_i \text{ as is})$

Reduction from Partition: $p = 2$ \Rightarrow \text{weakly NP-hard}

Reduction from 3-Partition: $p = \lceil n/3 \rceil$ \Rightarrow \text{strongly NP-hard}

(This was Garey & Johnson's motivation for introducing 3-partition in 1975.)

Claim: jobs finishable in makespan $\leq t$ \iff (3-Partition instance has a solution)

\[
\text{target sum}
\]
Rectangle packing: 
- given \( n \) rectangles & target rectangle \( A \rightarrow B \)
- can you pack former into latter? 
  \( \Rightarrow \) rotate & translate to fit without overlap 

- **OPEN**: \( \in \) NP? 
- special case: exact packing — no gaps 
  \( \Rightarrow \) hardness result is stronger theorem 
  - rotation \( \in \{0, 90^\circ, 180^\circ, 270^\circ\} \), translation integral 
    (proof by induction: consider corner, repeat) 
- NP certificate: translations & rotations

**Reduction from Partition:** 
\[
A = \begin{array}{cccc}
\alpha_1 & \alpha_2 & \cdots & \alpha_n \\
\end{array} \\
B = \begin{array}{c}
\epsilon \\
\end{array} \\
\text{avoid rotation:} \quad t = \epsilon \alpha_i / 2 \\
\text{avoid rotation:} \quad t = \epsilon \alpha_i / \sqrt{3}
\]

**Reduction from 3-Partition** 
\[
B = \begin{array}{c}
\frac{\alpha_i}{3} \\
\end{array} \\
\text{avoid rotation:} \quad t = \epsilon \alpha_i / (\sqrt{3})
\]

**Scaling trick** to make all dimensions integral: 
\[
A = \begin{array}{c}
\frac{n \alpha_i}{3} + 1 \\
\end{array} \\
B = \begin{array}{c}
\frac{n}{3} \\
\end{array}
\]

Here, just adding \( \frac{n}{3} \) to each \( \alpha_i \) suffices: 
\[
A = \begin{array}{c}
\frac{n \alpha_i}{3} + 1 \\
\end{array} \\
B = \begin{array}{c}
\frac{t + n}{3} \\
\end{array}
\]

[Demaine & Demaine - G&C 2007]
Edge-matching puzzles: [Demaine & Demaine - G&C 2007]
- given unit square tiles, each side labeled with a “color”
- given target rectangle
- goal: put tiles in target such that tiles sharing an edge have matching colors

No numbers ⇒ can’t use Partition!

Reduction from 3-Partition: (like rect. packing)
- a_i gadget:
  - effectively unary encoding!
  - prevents rotation
- if i colors go together, forced to make this
- but some could go on boundary...
- frame gadget:
  (“infrastructure”)

- unique colors forced on boundary
  ⇒ frame construction forced
- target shape: \((\lceil \frac{n}{3} \rceil + 2) \times (t+2)\)
  ⇒ a_i construction forced (no boundary left)
  ⇒ effectively rectangle packing
Signed edge-matching puzzles: (lizards etc.)
- colors come in matching pairs: a & A, b & B, etc.
- color does not match itself—only its mate

Reduction from unsigned edge-matching puzzles:

- interior colors (x, y, z, w) are unique pairs
  ⇒ must assemble 2x2
  (assuming frame to prevent boundary use)
  ⇒ acts like unsigned tile
Jigsaw puzzles: [Demaine & Demaine - G&C 2007]
- no guiding picture
- ambiguous mates (fitting ≠ correct)

Reduction from signed edge-matching puzzles:
\[
\begin{array}{c}
A \\
\rightarrow \\
C
\end{array} \quad \Rightarrow 
\begin{array}{c}
a \\
\rightarrow \\
6 \text{ shape}
\end{array}
\quad \text{lower case} \rightarrow \text{pocket}
\quad \text{upper case} \rightarrow \text{tab}

- for rectangular boundary:
\[
\text{Given square} \quad \Rightarrow 
\]

Polyomino packing: [Demaine & Demaine - G&C 2007]
- given polyominoes = edge-to-edge joinings (like Tetris)
- given target rectangle
- goal: exact pack former into latter

- rectangle packing is a special case ⇒ done
- but piece areas are \( > n \)
- what if areas are polylog?
- \( \text{OPEN} \): logarithmic area

Reduction from jigsaw puzzles:
\[
\begin{array}{c}
\text{\{binary encoding of color\}} \\
\rightarrow \\
(\log n)
\end{array}
\quad - \text{can get equal areas}
Closing the loop: reduction from polyomino packing to unsigned edge-matching puzzles

- use frame, but with $\$ = \$.

So: all 4 puzzle types are NP-complete & constant-factor equivalent: can convert one to the other with $O(1)$ factor blowup.

3-partition  

polyomino packing  

jigsaw

unsigned edge matching  

signed edge matching
Packing squares into a square: strongly NP-complete
[Leung, Tam, Wong, Young, Chin - JPDC 1990]
- motivation: scheduling square jobs on grid supercomputer

Rectangle target:
- squares of dimension $a_i + B < \text{huge} \Rightarrow \sim B$
- pack into rectangle of height $\sim 3B$:
  \[
  (B+t) \frac{n}{3}
  \]
- total slope $\leq (3B+t) \cdot (t \frac{n}{3})$
  $< B^2 < \text{one square}$
  if $B > t \cdot n$
  $\Rightarrow$ “doesn’t help”

Exact packing: add $1 \times 1$ \(\square\)s to fill extra area

Square target:
- infrastructure to build rectangular space

- scale by $3B+t$
- set $x$ large enough to get enough width
- pad excess with $B \times B$ squares
6.890 Algorithmic Lower Bounds: Fun with Hardness Proofs
Fall 2014

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.