Knapsack Problem

Instructor: Kwei-Long Huang

Course No: 546 U6110

Agenda

- Linear Programming Relaxation
- Dynamic Programming
- Lagrangian Multiplier Methods
- Network Approaches
- Applications and Uses of Knapsack
- Reducing Integer Programs to Knapsack

Relax Integer Constraints

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} v_j x_j \\ \text{s.t.} \quad & \sum_{j=1}^{n} w_j x_j \le b \\ & x_j \ge 0 \text{ for all } j \end{aligned}$$

- Let $y_j = w_j x_j$

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} (v_j / w_j) y_j \\ \text{s.t.} \quad & \sum_{j=1}^{n} y_j \le b \\ & y_j \ge 0 \text{ for all } j \end{aligned}$$

Linear Programming

- Assume the variables have been reordered:

$$v_1 / w_1 \le v_2 / w_2 \le \cdots \le v_n / w_n$$

- The LP optimal solution

$$y_n = b,\ x_n = b / w_n;$$
$$Z = (v_n / w_n) b$$

- Upper bound?
- Lower bound?
- The similar result when the variables have bounds.
  - Pack item n up to its bound, and followed by packing item $n-1$, and so on.

Agenda

- Linear Programming Relaxation
- Dynamic Programming
- Lagrangian Multiplier Methods
- Network Approaches
- Applications and Uses of Knapsack
- Reducing Integer Programs to Knapsack

Dynamic Programming

- DP is an optimization procedure that converts a problem with multiple decisions into a sequence of interrelated decisions.
- Stage (n): the original problem is divided into N stages. There is an initial stage and a terminating stage.
- State (s_n): each stage has a number of states associated with it. The states are various possible conditions in which the system might be at each particular stage of the problem.
- Decision variable (x_n): there is one decision variable for each stage of the problem.
- Optimal decision or policy($x_n^*(s_n)$): the optimal decision at a particular stage depends on the state. The DP procedure is designed to find an optimal decision at each stage for all possible state.

Dynamic Programming (con't)

- Optimal value or objective function ($f_n^*(s_n)$): best total value from stage n to the end, given that the starting state at stage n is s_n, and a sequence of optimal decision is made.

  $$f_n^*(s_n) = \max_{x_n}\{f_n(s_n, x_n)\}$$

  where $f_n(s_n,x_n)$ is total value from stage n to the end.
- Recursive relationship: identifies the optimal policy at stage n, given that the optimal policy at stage $(n+1)$ is available.

Algorithm 1

- Assumption:
  - The weight (w_i) are positive integers.
- Formulation
  - Let $f(w)$ = maximum value of the items with capacity w,

    $w_{\min} = \min\{w_i\}$, and v_i = value of one item of type i.
  - Initialization

$$f(w) = -\infty, \; w < 0;$$
$$f(w) = 0, \; w = 0, 1, \ldots, w_{\min} - 1.$$

Algorithm 1 (con't)

- Compute $f(w)$, $w = w_{\min}, w_{\min}+1, \ldots, W$

$$f(w) = \max_{i=1,\ldots,n} \{v_i + f(w - w_i)\}.$$

- Answer:

$$f(W)$$

Example (1/3)

- N=4, W=13.

Type (i)	Weight (w_i)	Value (v_i)	v_i / w_i
1	7	14	2.00
2	4	6	1.50
3	6	13	2.17
4	8	17	2.13

Example – Solution (2/3)

$f(w) =$

$f(w) =$

$f(6) =$

$f(7) = \max\ \{\mathbf{v_1 + f(7 - w_1)},\ v_2 + f(7 - w_2),\ v_3 + f(7 - w_3)\} = \max\ \{14, 6, 13\} = 14.$

$f(8) = \max\ \{v_1 + f(8 - w_1),\ v_2 + f(8 - w_2),\ v_3 + f(8 - w_3),\ \mathbf{v_4 + f(8 - w_4)}\}$
$= \max\ \{14+0, 6+6, 13+0, 17+0\} = 17.$

$f(9) = \max\ \{v_1 + f(9 - w_1),\ v_2 + f(9 - w_2),\ v_3 + f(9 - w_3),\ \mathbf{v_4 + f(9 - w_4)}\}$
$= \max\ \{14+0, 6+6, 13+0, 17+0\} = 17.$

$f(10) = \max\ \{v_1 + f(10 - w_1),\ \mathbf{v_2 + f(10 - w_2)},\ \mathbf{v_3 + f(10 - w_3)},\ v_4 + f(10 - w_4)\}$
$= \max\ \{14+0, 6+13, 13+6, 17+0\} = 19.$

$f(11) = \max\ \{\mathbf{v_1 + f(11 - w_1)},\ \mathbf{v_2 + f(11 - w_2)},\ v_3 + f(11 - w_3),\ v_4 + f(11 - w_4)\}$
$= \max\ \{14+6, 6+14, 13+6, 17+0\} = 20.$

$f(12) = \max\ \{v_1 + f(12 - w_1),\ \mathbf{v_2 + f(12 - w_2)},\ v_3 + f(12 - w_3),\ v_4 + f(12 - w_4)\}$
$= \max\ \{14+6, 6+17, 13+13, 17+6\} = 26.$

$f(13) = \max\ \{\mathbf{v_1 + f(13 - w_1)},\ v_2 + f(13 - w_2),\ \mathbf{v_3 + f(13 - w_3)},\ v_4 + f(13 - w_4)\}$
$= \max\ \{14+13, 6+17, 13+14, 17+6\} = 27.$

Example – Solution (3/3)

- The optimal solution can be obtained by backtracking.
- The same optimal solution is back-tracked in two different ways.

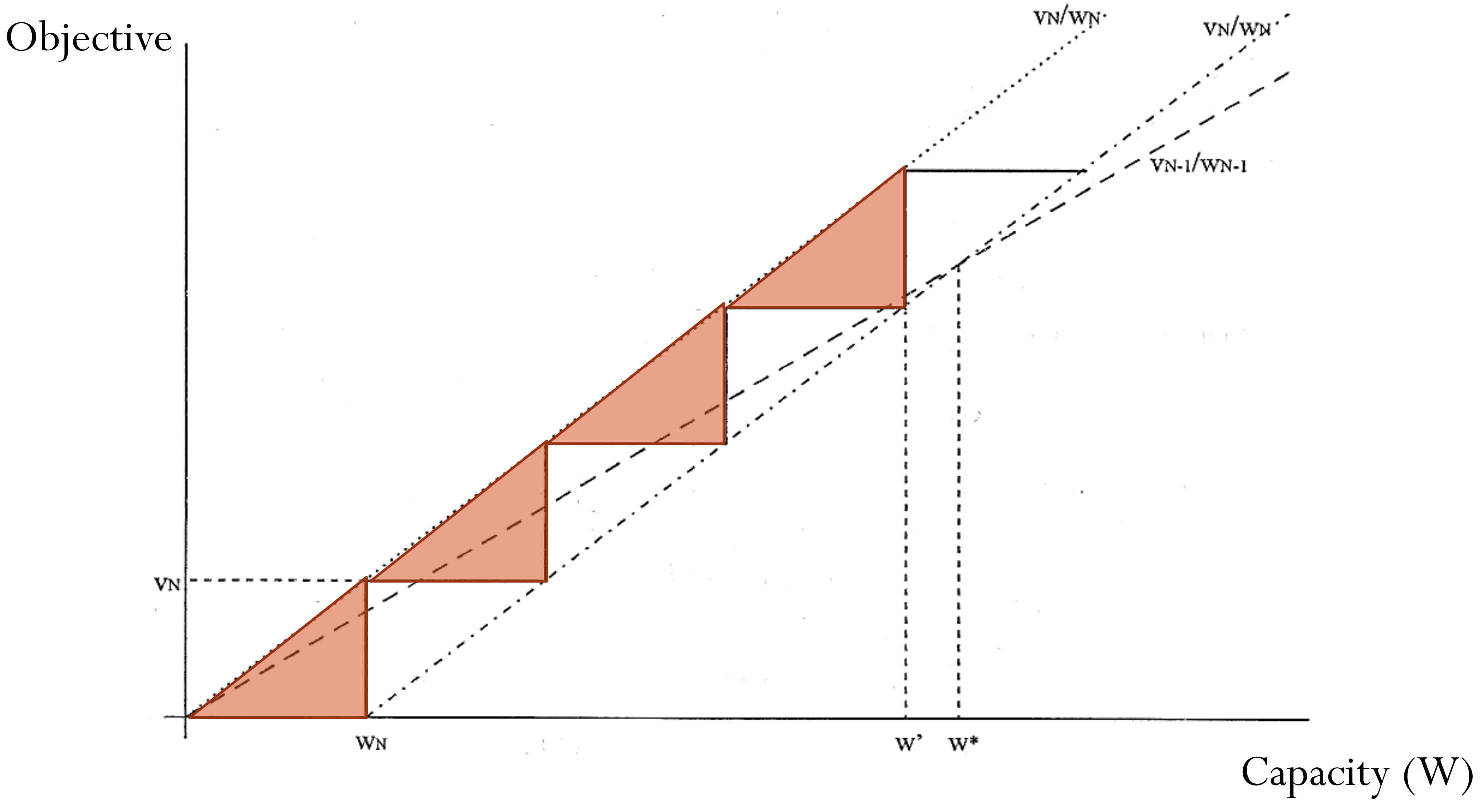
$$f(13) = 27, x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0.$$

- What if W is a large number?

Improvement

Rearrange items such that

$$\frac{v_1}{w_1} \le \frac{v_2}{w_2} \le \ldots \le \frac{v_N}{w_N}.$$



Observations

- The dotted line is to the optimal value. It is an optimal value if W/w_N is integer.
- The solid line is the optimal value. It corresponds to a feasible solution, i.e., $\left\lfloor W/w_N \right\rfloor$.
- The dashed line has slope v_{N-1}/w_{N-1}.
  - Which is the upper bound for the problem without item type N.
- w' is such that, for all w w', item type N is contained in the optimal solution.

Observations (con't)

- Dashed-dot line: $v = \left(\frac{v_N}{w_N}\right)(w - w_N)$. In the example, $v = \left(\frac{13}{6}\right)(w - 6)$. (Type 3).

  Dashed line: $v = \left(\frac{v_{N-1}}{w_{N-1}}\right)w$. In the example, $v = \left(\frac{17}{8}\right)w$. (Type 4).

  Intersection (w*): Example, $w^* = \frac{13}{\left(\frac{13}{6}\right) - \left(\frac{17}{8}\right)} = 312$.

  Value of w': w' . In the example,

- w'=312 is still a larger number.
- The smallest weight w" such that for all w ≥ w" item N is in the optimal solution may be considerably less than w'.

The Most Valuable Item

- **Theorem**: item type N is in all optimal solution for all w≥ w", item type N is part of every optimal solution for w", w"+1, …, w"+w_{max}− 1.
- The optimal solution for any weight capacity W larger than w" is found by first determining the smallest integer *k* such that

$$W - kw_N \Rightarrow k = \left\lceil \frac{W - w''+1}{w_N} \right\rceil.$$

- Take *k* items of type N and solve the problem with weight capacity W − kw_N
- The example in page 10, w"=18.

Agenda

- Linear Programming Relaxation
- Dynamic Programming
- Lagrangian Multiplier Methods
- Network Approaches
- Applications and Uses of Knapsack
- Reducing Integer Programs to Knapsack

Lagrangian Relaxation

- Consider the integer problem

$$\begin{aligned} &\max \ \mathrm{cx} \\ &\text{s.t.} \quad Ax \le b \\ &\qquad\quad x \ge 0 \text{ and integer.} \end{aligned}$$

- Let λ be an (m) column of nonnegative numbers (multipliers)

$$\begin{aligned} &\max \ cx - \lambda Ax \\ &\text{s.t.} \quad x \ge 0 \text{ and integer.} \qquad (L) \end{aligned}$$

Lagrangian Relaxation

- If x^0 solves problem (L), then it also solves the integer program with b replaced by Ax^0.
- Therefore, if λ is chosen so that the optimal solution x^0 gives $b=Ax^0$, the original problem has been solved.
- $b=Ax^0$ is usually easier to solve. The difficulty is to find that multiplier λ which gives the equality.

Relax One Constraint

- Problem L for the knapsack problem

$$\begin{aligned} \max \quad & \textstyle\sum_{j=1}^{n} c_j x_j - \lambda \sum_{j=1}^{n} a_j x_j + \lambda b \\ \text{s.t.} \quad & x_j \geq 0 \text{ and integer for all } j \end{aligned}$$

$$\begin{aligned} \max \quad & \textstyle\sum_{j=1}^{n} (c_j - \lambda a_j) x_j + \lambda b \\ \text{s.t.} \quad & x_j \geq 0 \text{ and integer for all } j \end{aligned}$$

Special Solution Method

- By inspection

$$x_j^0 = u_j \quad \text{if } c_j - \lambda a_j \qquad (u_j \text{ is an integer upper bound for } x_j)$$

$$x_j^0 = t \quad \text{if } c_j - \lambda a_j \qquad (t \text{ is any integer satisfying } 0 \le t \le u_j)$$

$$x_j^0 = 0 \quad \text{if } c_j - \lambda a_j$$

- As x^0 changes only when , the value for x^0 remains the same in the intervals (item n is the most valuable)

$$0 \le \lambda < c_1/a_1,\ c_1/a_1 \le \lambda < c_2/a_2, \ldots,\ c_{n-1}/a_{n-1} \le \lambda < c_n/a_n,\ c_n/a_n \le \lambda$$

Example (1/3)

- Consider the following BIP

$$\begin{aligned} \text{maximize}\quad & 4x_1 + 8x_2 + 14x_3 + 18x_4 \\ \text{subject to}\quad & 4x_1 + 7x_2 + 12x_3 + 15x_4 \le 33 \\ & x_j = 0 \text{ or } 1. \end{aligned}$$

- Problem L

$$\begin{aligned} \text{maximize}\quad & (4-4\lambda)x_1 + (8-7\lambda)x_2 + (14-12\lambda)x_3 + (18-15\lambda)x_4 \\ \text{subject to}\quad & x_j = 0 \text{ or } 1. \end{aligned}$$

Example – Solution (2/3)

- Determine the interval of λ

  $\lambda =$

λ	Sign of $c_j - \lambda a_j$				Value				Volume	Slack	Obj.
	j =1	j =2	j =3	j =4	x_1	x_2	x_3	x_4	$\sum a_j x_j$	$b - \sum a_j x_j$	$\sum c_j x_j$
0	+	+	+	+	1	1	1	1	38	-5	44
1	X	+	+	+	$\underline{\mathbf{0}}$	1	1	1	34	-1	40
					$\underline{\mathbf{1}}$	1	1	1	38	15	44
8/7	-	X	+	+	0	$\underline{\mathbf{0}}$	1	1	27	6	32
					0	$\underline{\mathbf{1}}$	1	1	34	-1	40
14/12	-	-	X	+	0	0	$\underline{\mathbf{0}}$	1	15	18	18
					0	0	$\underline{\mathbf{1}}$	1	27	6	32
18/15	-	-	-	X	0	0	0	$\underline{\mathbf{0}}$	0	33	0
					0	0	0	$\underline{\mathbf{1}}$	15	18	18

Example – Solution (3/3)

- There doesn't exist a λ for which $b - \sum_{j=1}^{4} a_j x_j^0 = 0$.

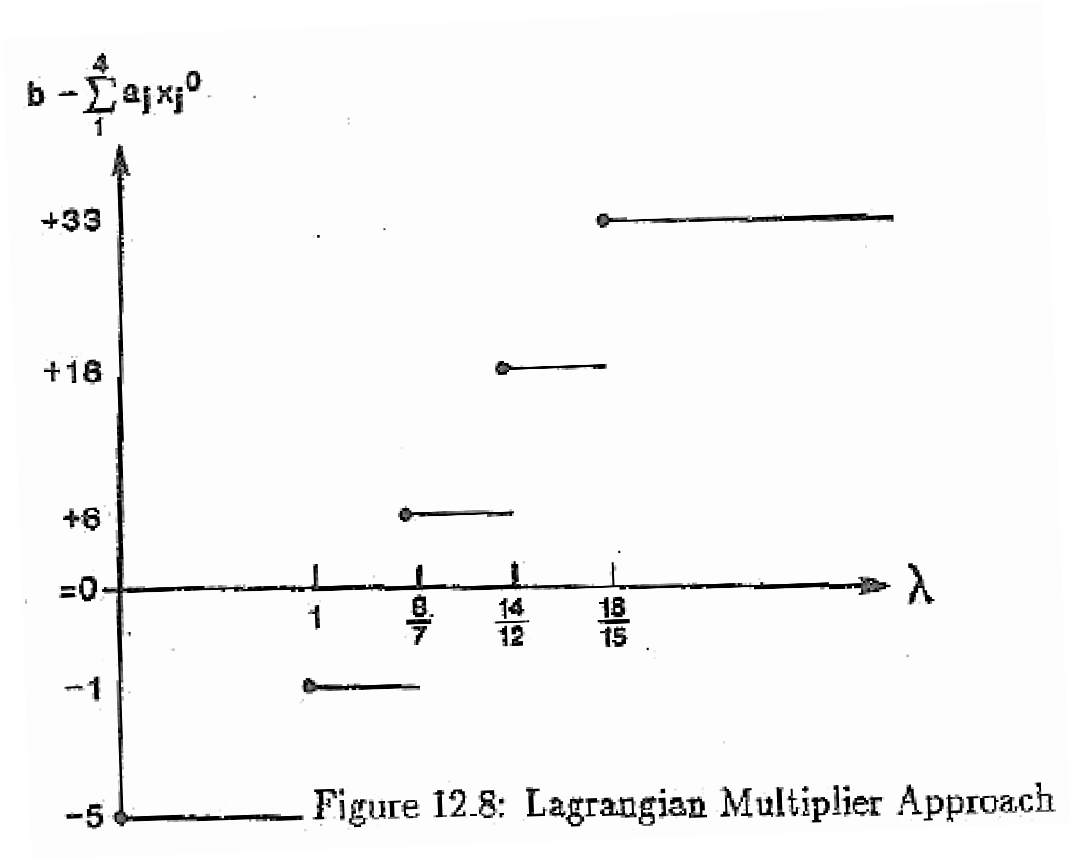


Figure 12.8: Lagrangian Multiplier Approach

Agenda

- Linear Programming Relaxation
- Dynamic Programming
- Lagrangian Multiplier Methods
- Network Approaches
- Applications and Uses of Knapsack
- Reducing Integer Programs to Knapsack

Shortest Path

- A knapsack problem with an equality constraint can be transformed to a shortest path problem.
- The network is with $b+1$ nodes
- For item j, there are arcs for every pair of nodes (g, t) where node $g(N_g)$ – node $t(N_t) = a_j$ and the associated costs are $-c_j$.
- The source node is N_0 and the destination node is N_b.
- If b is too large and (or) a_j is relatively small, the approach becomes inefficient.

Example

- Consider the following problem

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 2x_2 \\ \text{subject to} \quad & 4x_1 + 2x_2 + x_3 = 8 \\ & x_j \geq 0 \text{ and integer.} \end{aligned}$$

- The network is

Example (con't)

- The shortest path is

$$\begin{tikzcd} 0 \arrow[r, "-2"] & 2 \arrow[r, "-2"] & 4 \arrow[r, "-2"] & 6 \arrow[r, "-2"] & 8 \end{tikzcd}$$

- That is, $x_2=4$, $x_1=x_3=0$.

Agenda

- Linear Programming Relaxation
- Dynamic Programming
- Lagrangian Multiplier Methods
- Network Approaches
- Applications and Uses of Knapsack
- Reducing Integer Programs to Knapsack

Knapsack Problem

- Consider the integer program with a single constraint

$$\begin{aligned} \text{max} \quad & \textstyle\sum_{j=1}^{n} c_j x_j \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{n} a_j x_j \le b \\ & x_j \ge 0 \text{ and integer for all } j \end{aligned}$$

Where the costs (values) c_j, coefficients a_j and RHS b are integer.

- Packing a knapsack so that its capacity is not exceeded and the total value is maximized.

Related Applications

- It is representative of several practical situations
  - Capital budgeting
  - Project selection
  - Loading problem
  - Capital investment
- It appears as a subproblem that has to be solved in many integer programming algorithms

Capital Budgeting

- Choosing among *n* competing investment possibilities so as to maximized the total payoff subject to limited funds.

$$
\begin{aligned}
\text{max} \quad & \sum_{j=1}^{n} c_j x_j \\
\text{s.t.} \quad & \sum_{j=1}^{n} a_j x_j \le b \\
& x_j = 0 \text{ or } 1 \text{ for all } j
\end{aligned}
$$

Multiperiod Capital Budgeting

- Investment over several periods

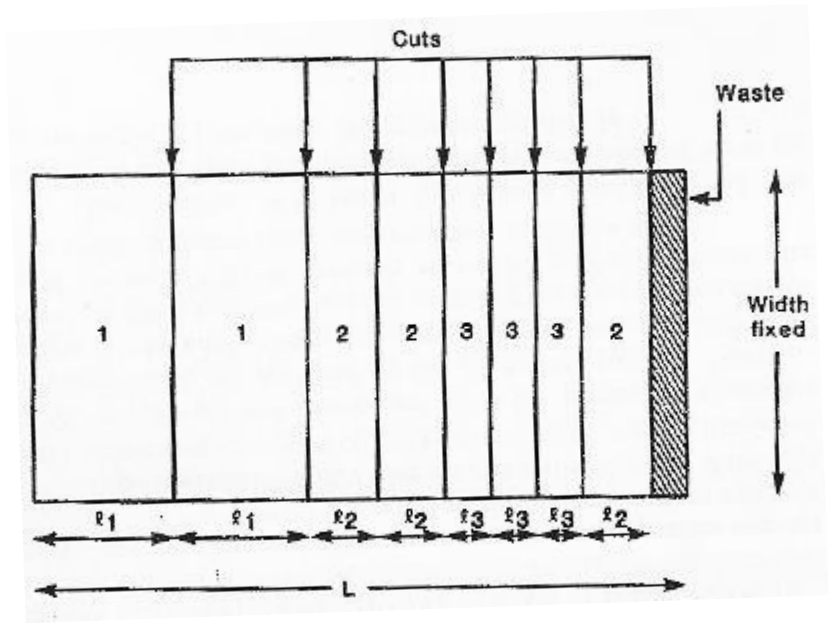
$$\begin{aligned}\text{max} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{s.t.} \quad & \sum_{j=1}^{n} a_{tj} x_j \le b_t \\ & x_j = 0 \text{ or } 1 \text{ for all } j\end{aligned}$$

- Multiple choice constraints: *n* investments are partitioned into disjoint sets n_p

$$\sum_{j \in n_i} x_j = 1, \; i = 1, \dots, p$$

Cutting Stock Problem

- Each standard length or roll is to be sliced into lengths l_i $(i=1,\ldots,m)$
- Cut up rolls of the material so that the demand for the number of pieces of lengths l_i is satisfied while the usage of rolls is minimized.



Cutting Stock Problem

- Let N_i be the number of pieces of lengths l_i needed, c_j be the cost of the roll from which *jth* cutting pattern is cut, and a_{ij} be the number of pieces of length l_i produced while using *jth* cutting pattern.
- Let x_j be the number of times the *jth* cutting pattern is used.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} c_j x_j \\ \text{s.t.} \quad & \sum_{j=1}^{n} a_{ij} x_j \geq N_i \;, \; i = 1, \ldots, m \\ & x_j \geq 0 \text{ and integer for all } j \end{aligned}$$

Loading Problems

- A fleet of *m* trucks carrying various items.
- Given *n* indivisible items.
- Let x_{ij} be a indicator which represents item *j* is carried by truck *i*

$$
\begin{aligned}
\max \quad & \sum_{j=1}^{n} c_j x_{ij} \\
\text{s.t.} \quad & \sum_{j=1}^{n} a_j x_{ij} \le b_i \quad , i = 1, \ldots, m \\
& \sum_{j=1}^{n} x_{ij} \le 1 \qquad , j = 1, \ldots, n \\
& x_{ij} = 0 \text{ or } 1
\end{aligned}
$$

Change Making Problem

- Suppose there are n types of coins, where each type j has denomination w_j. A cashier wishes to make change to meet a given amount b using the least number of coins.
- Let x_j be the number of coins j selected.

Agenda

- Linear Programming Relaxation
- Dynamic Programming
- Lagrangian Multiplier Methods
- Network Approaches
- Applications and Uses of Knapsack
- Reducing Integer Programs to Knapsack

Aggregating Constraints

- A system of linear equations with integer coefficients can be transformed to a single linear equation.
- These two problems have the same set of nonnegative integer solutions.
- Consider the *m* linear equations:

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \ , i = 1, \ldots, m \ (*)$$

- Find weights w_i so that every nonnegative integer solution to the single constraint is also a solution to (*).

$$\sum_{j=1}^{n} a_j x_j = b \text{ where } a_j = \sum_{i=1}^{m} w_i a_{ij} \text{ and } b = \sum_{i=1}^{m} w_i b_i \ (**)$$

Aggregating Constraints

- For arbitrary weights, the set of nonnegative integers x satisfying (**) is usually **larger** than the set satisfying (*).
- Example

$$\begin{array}{llr}
\text{minimize} & 5x_1 + 7x_2 + 10x_3 + 3x_4 + x_5 & \\
\text{subject to} & -x_1 + 3x_2 - 5x_3 - x_4 + 4x_5 + x_6 = -2, & \text{(i)} \\
 & 2x_1 - 6x_2 + 3x_3 + 2x_4 - 2x_5 + x_7 = 0, & \text{(ii)} \\
 & x_2 - 2x_3 + x_4 + x_5 + x_8 = -1, & \text{(iii)} \\
 & x_j = 0 \text{ or } 1 & (j=1, 2, 3, 4, 5) \\
\text{and} & x_j \geq 0 \text{ and integer} & (j=6, 7, 8)
\end{array}$$

Example

- The solutions to the problem are (0,1,1,0,0) and (1,1,1,0,0).
- Set $w_1=w_2=w_3=1$, the resulted equation is:
- The two solutions included? And what else?

An Aggregation Process (Mathews)

- Theorem: consider a system of two linear equations

$$s_1 \equiv \sum_{j=1}^{n} a_{1j} x_j = b_1, \quad (i)$$

$$s_2 \equiv \sum_{j=1}^{n} a_{2j} x_j = b_2, \quad (ii) \text{ where } a_{ij} > 0 \text{ and integer}$$

(a) If there exists a nonnegative solution to the system, then

$$b_2 a_{1j} / a_{2j} \geq b_1, \quad \text{for at least one } j$$

(b) If w is any positive integer such that $w > b_2 \max_j \{a_{1j} / a_{2j}\}$.

Then, the solution set of the system is the same as that of the single equation

$$s_1 + w s_2 = b_1 + w b_2$$

An Aggregation Process

- Generate a pair of equations as follows:
  $s_1 + s_2 = b_1 + b_2$ and $s_1 + 2s_2 = b_1 + 2b_2$
- Pairwise aggregate two equations until a single one is left.
- Negative coefficient: if $a_{1j} < 0$ and $x_j \leq u_j$ (positive integral upper bound), then the system is convertible.
- Let $\bar{x}_j = u_j - x_j$.
- Any integer program which has a bounded linear programming feasible region with at least one integer point can be transformed to an equivalent knapsack problem.

Example (1/3)

$$\begin{aligned}
-x_1 + 3x_2 - 5x_3 - x_4 + 4x_5 + x_6 \quad &= -2, \quad \text{(i)} \\
2x_1 - 6x_2 + 3x_3 + 2x_4 - 2x_5 \quad + x_7 \quad &= 0, \quad \text{(ii)} \\
x_2 - 2x_3 + x_4 + x_5 \quad + x_8 &= -1, \quad \text{(iii)} \\
x_j = 0 \text{ or } 1 \quad & (j=1, 2, 3, 4, 5)
\end{aligned}$$

- Transfer all equations with positive coefficients
- Take (*i*) for example, replace x_1 by $1 - \bar{x}_1$, x_3 by $1 - \bar{x}_3$, and x_4 by $1 - \bar{x}_4$.

Example (2/3)

- The initial system with all positive coefficients

Eq./Variable	x_1	$\overline{x}_1$	x_2	$\overline{x}_2$	x_3	$\overline{x}_3$	x_4	$\overline{x}_4$	x_5	$\overline{x}_5$	x_6	x_7	x_8	b
(i)’		1	3			5		1	4		1			5
(ii)’	2			6	3		2			2		1		8
(iii)’			1			2	1			1			1	1

- Generate a pair of equations

	x_1	$\overline{x}_1$	x_2	$\overline{x}_2$	x_3	$\overline{x}_3$	x_4	$\overline{x}_4$	x_5	$\overline{x}_5$	x_6	x_7	x_8	b
	2	1	3	6	3	5	2	1	4	2	1	1		13
	4	1	3	12	6	5	4	1	4	4	1	2		21

- What is the value of w?

Example (3/3)

- Set w=22, the following system is yielded.

	x_1	$\overline{x}_1$	x_2	$\overline{x}_2$	x_3	$\overline{x}_3$	x_4	$\overline{x}_4$	x_5	$\overline{x}_5$	x_6	x_7	x_8	b
(i)''=(i)+22(ii)	90	23	69	270	135	115	90	23	92	90	23	45		475
(ii)''=(iii)'			1			2	1		1				1	1

- Again, generate a pair of equations: (i)"+(ii)" and (i)"+2(ii)".
- We find that w >477, then set w =478.
- The aggregation equation:

$$\begin{aligned}43110x_1+11017\overline{x}_1+34008x_2+129330\overline{x}_2+64665x_3\\+56999\overline{x}_3+44067x_4+11017\overline{x}_4+45025x_5\\+43110\overline{x}_5+11017x_6+21555x_7+957x_8=228482\end{aligned}$$

or

$$\begin{aligned}32093x_1-95322x_2+7666x_3+33050x_4+1915x_5\\+11017x_6+21555x_7+957x_8=-22991\end{aligned}$$

An Improved Aggregation Process (1/3)

- Consider two constraints and combine them into one

$$\sum_{j=1}^{n} d_j x_j = b_1,$$

$$\sum_{j=1}^{n} f_j x_j = b_2.$$

- Assume that d_j, f_j, b_1, and b_2 are integers.
- Each x_j has a bound u_j

- Let

$$\lambda^+ = \max\left\{\sum_{j=1}^{n} d_j x_j - b_1 : 0 \le x_j \le u_j, \text{integer}, j = 1, \ldots, n\right\},$$

$$= \sum_{j=1}^{n} \max\{0, d_j\} \cdot u_j - b_1$$

$$\lambda^- = \min\left\{\sum_{j=1}^{n} d_j x_j - b_1 : 0 \le x_j \le u_j, \text{integer}, j = 1, \ldots, n\right\},$$

$$= \sum_{j=1}^{n} \min\{0, d_j\} \cdot u_j - b_1$$

$$\lambda = \max\left\{\left|\sum_{j=1}^{n} d_j x_j - b_1\right| : 0 \le x_j \le u_j, \text{integer}, j = 1, \ldots, n\right\},$$

$$= \max\{\lambda^+, |\lambda^-|\}$$

An Improved Aggregation Process (2/3)

- **Theorem:** The integer vector x^0, $0 \leq x^0 \leq u$, is a solution to the two equations if and only if

$$\sum_{j=1}^{n} \left(d_j + \alpha f_j \right) x_j^0 = b_1 + \alpha b_{2,} \quad (3)$$

where α is any integer satisfying $|\alpha| > \lambda$.

- Proof:

(=>) given x^0 is the solution to the two equations, to show x^0 is also a solution to (3)

An Improved Aggregation Process (3/3)

(<=) Suppose that x^0 solves (3) and $\sum_{j=1}^{n} f_j x_j^0 = b_2 + k,$ (4) where k is an arbitrary integer. It will be shown that $|\alpha| > \lambda \Rightarrow k = 0$.

- Multiply (4) by α and subtract the result from (3).

$$\sum_{j=1}^{n} d_j x_j^0 = b_1 - k\alpha$$

- Then,

$$|\alpha| > \lambda \geq \qquad\qquad = |-k\alpha| = |k||\alpha|$$

$$\Rightarrow \quad |\alpha| > |k||\alpha| \quad \Rightarrow \quad |k| < 1 \xrightarrow[k \text{ integer}]{} k = 0$$

$$\Rightarrow \sum_{j=1}^{n} f_j x_j^0 = b_2 \text{ and } \sum_{j=1}^{n} d_j x_j^0 = b_1.$$

Example (1/4)

$$\begin{array}{llcll}
\text{maximize } z = 2x_1 + x_2 & & & & \\
\text{subject to} & x_1 + x_2 \le 5 & \Rightarrow & x_1 + x_2 + x_3 = 5, & (1) \\
& -x_1 + x_2 \le 0 & \Rightarrow & -x_1 + x_2 + x_4 = 0, & (2) \\
& 6x_1 + 2x_2 \le 21 & \Rightarrow & 6x_1 + 2x_2 + x_5 = 21, & (3) \\
& x_1, x_2 \ge 0, \text{integer} & \Rightarrow & x_1, x_2, x_3, x_4, x_5 \ge 0, \text{integer}. &
\end{array}$$

- The upper bound for each variable

Example (2/4)

- Consider constraint (3)

$$For\ 6x_1 + 2x_2 + x_5 = 21,$$

$$\lambda^+ =$$

$$\lambda^- =$$

$$\lambda = \max\left\{ \lambda^+, \left| \lambda^- \right| \right\} = 24$$

- Combine (2) and (3)

$$6x_1 + 2x_2 + x_5 + \alpha\left(-x_1 + x_2 + x_4\right) = 21 + \alpha \cdot 0,\quad for\ \ |\alpha| > 24,$$

$$\alpha = 25 \Rightarrow -19x_1 + 27x_2 + 25x_4 + x_5 = 21.\ \ (*)$$

Example (3/4)

- For constraint (1)

For $x_1 + x_2 + x_3 = 5,$

$$\lambda^+ = 1\times 3 + 1\times 3 + 1\times 5 - 5 = 6,$$

$$\lambda^- = 0\times 3 + 0\times 3 + 0\times 5 - 5 = \text{-}5,$$

$$\lambda \quad = \max\left\{ \lambda^+, \left| \lambda^- \right| \right\} = 6.$$

- Combine (1) and (*)

$$x_1 + x_{2+} x_3 + \alpha\left(-19x_1 + 27x_2 + 25x_4 + x_5\right) = 5 + \alpha \cdot 21, \;\; \mathit{for} \;\; |\alpha| > 6,$$

$$\alpha = 7 \;\Rightarrow\; -132x_1 + 190x_2 + x_3 + 175x_4 + 7x_5 = 152. \;\; (**)$$

Example (4/4)

Therefore,

$$\begin{aligned}
&\text{maximize } z = 2x_1 + x_2, \\
&\text{subject to } -132x_1 + 190x_2 + x_3 + 175x_4 + 7x_5 = 152. \\
&x_1' = 3 - x_1 \qquad (x_1 = 3 - x_1').
\end{aligned}$$

$$\begin{aligned}
\Rightarrow\quad &\text{maximize z} = -2x_1' + x_2 + 6 \\
&\text{subject to } 132x_1' + 190x_2 + x_3 + 175x_4 + 7x_5 = 548.
\end{aligned}$$

$$\begin{aligned}
&0 \le x_1' \le 3, \text{ integer}, \\
&0 \le x_2 \le 3, \text{ integer}, \\
&0 \le x_3 \le 5, \text{ integer}, \\
&0 \le x_4 \le 3, \text{ integer}, \\
&0 \le x_5 \le 21, \text{ integer}.
\end{aligned}$$

Reminder

- Homework 5 due on 6/1
- Final Project (6/15)(15% of your final grade)
  - Please prepare a 20-min presentation.
  - Every student should present.
  - Also submit a report with at most 10 pages before 6/26, and a draft is required on 6/15.
- Final exam on 6/22 (30%)