Algoritmos Evolutivos

Daniela Niño

Descripción del algoritmo

1. Se crea una población inicial
2. Se seleccionan dos elementos de la población inicial
3. Se hace un cruce entre los dos elementos seleccionados
4. Se hace una mutación o una búsqueda local con VND según una probabilidad
5. Se actualiza la población
6. Se repite del paso 2 al 4 según la cantidad de hijos que se definan
7. Se repite el paso 6 hasta que se agote el tiempo límite de la instancia JSSP
8. Se retorna la mejor solución de la población

Descripción del algoritmo

1. Se crea una población inicial

Se usa el algoritmo de ruido del trabajo 1 para crear una cantidad (nsol) de soluciones iniciales.

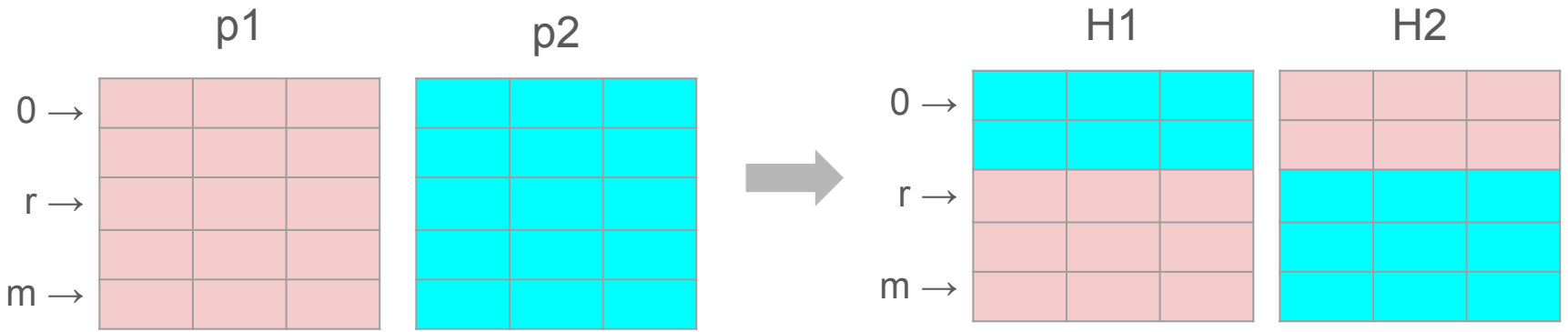
2. Se seleccionan dos elementos de la población inicial

Aleatoriamente se eligen dos soluciones iniciales (p1 y p2). La probabilidad de que una solución sea seleccionada aumenta entre menor sea su makespam y decrece entre mayor sea su makespam.

Descripción del algoritmo

3. Se hace un cruce entre los dos elementos seleccionados

Se selecciona un número (r) entero aleatorio entre [2, m]. Se crean dos nuevas matrices (H1 y H2) usando el orden de procesamiento de las máquinas [1, r] de p1 y [r+1, m] de p2 y viceversa, así:



* m es la cantidad de máquinas

Descripción del algoritmo

3. Se hace un cruce entre los dos elementos seleccionados

Luego de haber creado H1 y H2 se verifica si estas son factibles.

- Si ambas son factibles se retorna una de ellas aleatoriamente
- Si solo una de ellas es factible se retorna esa solución
- Si ninguna es factible se elige otro r para crear dos nuevas soluciones

Descripción del algoritmo

4. Se hace una mutación o una búsqueda local con VND según una probabilidad

Existe una probabilidad de mutación y una probabilidad para hacer búsqueda local, siempre esta segunda probabilidad es mayor que la primera. Además, se selecciona un número aleatorio (rand) entre [0, 1].

- Si rand < probabilidad de mutación: se le hace una mutación a la solución
- Si rand < probabilidad de LS: se hace búsqueda local con VND a la solución
- En caso contrario no se modifica la solución

Descripción del algoritmo

5. Se actualiza la población

Se busca el papá con el makespam más grande y se reemplaza por el hijo que tenga el makespam más pequeño.

6. Se repite nc veces (cantidad de hijos) del paso 2 al 4

7. Se repite el paso 6 hasta que se agote el tiempo límite de la instancia

8. Se retorna la mejor solución de la población

Comparaciones con diferentes valores de parámetros

Avg f(s)	Avg gap	Parámetros
1780	0.45	nsol = 5; nc = 3; prob_muta = 0.2; prob_LS = 0.8
1775	0.45	nsol = 8; nc = 3; prob_muta = 0.2; prob_LS = 0.8
1781	0.45	nsol = 5; nc = 5; prob_muta = 0.2; prob_LS = 0.8
1786	0.46	nsol = 8; nc = 5; prob_muta = 0.2; prob_LS = 0.8

* Debido a que los algoritmos se demoran mucho tiempo en correr, solo se probaron los diferentes parámetros con unas instancias, no con todas.

Comparaciones con diferentes valores de parámetros

Avg f(s)	Avg gap	Parámetros
1774	0.45	nsol = 5; nc = 3; prob_muta = 0.4; prob_LS = 0.9
1770	0.44	nsol = 8; nc = 3; prob_muta = 0.4; prob_LS = 0.9
1782	0.45	nsol = 5; nc = 5; prob_muta = 0.4; prob_LS = 0.9
1778	0.45	nsol = 8; nc = 5; prob_muta = 0.4; prob_LS = 0.9

* Debido a que los algoritmos se demoran mucho tiempo en correr, solo se probaron los diferentes parámetros con unas instancias, no con todas.

Comparaciones con diferentes valores de parámetros

Avg f(s)	Avg gap	Parámetros
1777	0.45	nsol = 5; nc = 3; prob_muta = 0.2; prob_LS = 0.7
1775	0.45	nsol = 8; nc = 3; prob_muta = 0.2; prob_LS = 0.7
1775	0.45	nsol = 5; nc = 5; prob_muta = 0.2; prob_LS = 0.7
1775	0.45	nsol = 8; nc = 5; prob_muta = 0.2; prob_LS = 0.7

* Debido a que los algoritmos se demoran mucho tiempo en correr, solo se probaron los diferentes parámetros con unas instancias, no con todas.

Makespam del algoritmo

	JSSP1	JSSP2	JSSP3	JSSP4	JSSP5	JSSP6	JSSP7	JSSP8
f(x)	1394	1414	1713	1535	1975	1847	2134	2274
%	40.7%	43.6%	45.5%	26.0%	61.5%	45.4%	17.4%	23.7%

	JSSP9	JSSP10	JSSP11	JSSP12	JSSP13	JSSP14	JSSP15	JSSP16
f(x)	2371	2502	3157	3194	3383	3410	6092	5786
%	26.4%	40.2%	11.8%	6.4%	16.0%	15.6%	10.3%	1.9%

	f(x)	%
Avg	2761	27.0%

Comparación con algoritmos anteriores

	Const.	Noise	GRASP
Avg	36,2%	34,6%	37,4%

	IB-BI	IB-FI	IF-BI	IF-FI	I-BI	I-FI
Avg	32,3%	35,6%	31,5%	35,7%	31,6%	35,6%

	VND	Mixed
Avg	32.4%	30.8%

	GA
Avg	27.0%

* El método Noise con r = 5 y GRASP con k = 3.

Tiempo de cómputo

	JSSP1	JSSP2	JSSP3	JSSP4	JSSP5	JSSP6	JSSP7	JSSP8
seg	3	3	4	4	85	77	94	99

	JSSP9	JSSP10	JSSP11	JSSP12	JSSP13	JSSP14	JSSP15	JSSP16
seg	75	84	209	176	426	352	1808	1838

	Avg
seg	333

Conclusiones

- El gap no cambia significativamente al cambiar los valores de los parámetros, esto puede suceder porque el algoritmo está explorando una misma zona donde se está quedando estancado.
- El algoritmo genético usando VND con first improvement fue mejor que todos los algoritmos de los trabajos anteriores.
- El algoritmo genético logró encontrar una mejor solución que todos los algoritmos del trabajo 2. Si comparamos el mejor algoritmo del trabajo 2 (insertion forward, best improvement) con el algoritmo genético, podemos evidenciar una diferencia de 4.5% en el gap a favor del algoritmo genético, pero este se demoró corriendo 30 minutos más (0.34 veces más) que el algoritmo de best improvement.

Conclusiones

- El algoritmo genético logró encontrar una mejor solución que todos los algoritmos del trabajo 3 en la misma cantidad de tiempo, logrando una ventaja de 3.8% en el gap respecto al algoritmo mixto y de 5.4% respecto al VND.
- Cuando se usa el algoritmo de ruido (trabajo 1), entre más grande es la instancia, menor es el gap, pero cuando se usan los algoritmos de búsqueda local y el genético (trabajos 2, 3 y 4), entre más grande es la instancia, menos reducción hay en el gap. Esto se puede generar porque para las instancias grandes, al tener un gap muy cercano a cero, a los algoritmos de búsqueda local y el genético se les dificulta más encontrar mejores soluciones.

Referencias Bibliográficas

Rivera Agudelo, J. C. (2023). Algoritmos Genéticos [Diapositivas]. Escuela de Ciencias Aplicadas e Ingeniería, Universidad EAFIT. https://interactivavirtual.eafit.edu.co/