A Memetic Algorithm for Reconstructing Cross-Cut Shredded Text Documents

Christian Schauer[1], Matthias Prandtstetter[2], and Günther R. Raidl[1]

[1] Institute of Computer Graphics and Algorithms
Vienna University of Technology, Vienna, Austria
{schauer|raidl}@ads.tuwien.ac.at
[2] Mobility Department—Dynamic Transportation Systems
Austrian Institute of Technology, Vienna, Austria
matthias.prandtstetter@ait.ac.at

Abstract. The reconstruction of destroyed paper documents became of more interest during the last years. On the one hand it (often) occurs that documents are destroyed by mistake while on the other hand this type of application is relevant in the fields of forensics and archeology, e.g., for evidence or restoring ancient documents. Within this paper, we present a new approach for restoring cross-cut shredded text documents, i.e., documents which were mechanically cut into rectangular shreds of (almost) identical shape. For this purpose we present a genetic algorithm that is extended to a memetic algorithm by embedding a (restricted) variable neighborhood search (VNS). Additionally, the memetic algorithm's final solution is further improved by an enhanced version of the VNS. Computational experiments suggest that the newly developed algorithms are not only competitive with the so far best known algorithms for the reconstruction of cross-cut shredded documents but clearly outperform them.

1 Introduction

Although many sensitive documents are electronically prepared, transmitted and stored nowadays, it is still common and often necessary to print them—especially due to legal reasons. In many cases these (printed) documents are then stored in archives and after a while they are destroyed by either burning them or by using mechanical machines called shredders which cut them according to security standards either in thin strips or small rectangles. Although this process of destruction is performed for making the documents unreadable, it is in some situations desirable to reconstruct the destroyed documents, e.g., when destruction was performed by mistake or in forensics or archeology.

Within this work we focus on the reconstruction of cross-cut shredded text documents (RCCSTD), i.e., of documents which were cut into rectangles of identical shape. In this case it is not possible to gather any helpful information from the snippets' shapes that can be exploited during the reconstruction process. Hence, solely the information printed on the snippets can be utilized. Because information retrieval is mainly attached to the fields of pattern recognition and

image processing, we will not further examine this process. We lay our focus on the reconstruction part using a combinatorial optimization approach based on a simple but for our needs effective pattern recognition function. Nevertheless, through the modularity of our implementation this function can easily be replaced by more sophisticated techniques. Therefore we will present a memetic algorithm (MA) [14] for solving RCCSTD incorporating a variable neighborhood search (VNS) [9] as local improvement procedure.

Based on the formal definition in [16] RCCSTD can be expressed as follows: Given is the output of a cross-cut shredding device, i.e., a set $S = \{1, \ldots, n\}$ of rectangular, geometrically identical snippets. All blank shreds, i.e., shreds with no helpful information printed on them, are replaced by a single "virtual" blank shred. Without loss of generality let us assume that this blank shred is shred n. To simplify matters, we further assume that the orientation of all shreds is known, e.g., identified using methods like those presented in [1, 13]. Furthermore, we are given an error estimation function $c(i, j)$, for all $i, j \in S$, computing for each pair of shreds an estimated error made when placing shred i left to shred j. Analogously, an error estimation function $\widehat{c}(i, j)$ estimating the error when placing shred i on top of shred j is given. In our case, these error estimation functions solely rely on the pixel information along the touching edges of the shreds and compare them by measuring the differences in the concrete gray values. A detailed definition of $c(i, j)$ and $\widehat{c}(i, j)$ can be found in [16].

A solution to RCCSTD is an injection $\Pi : S \setminus \{n\} \to \mathbb{D}^2$ of shreds to positions $p = (x, y)$ in the two-dimensional (Euclidean) space, with $x, y \in \mathbb{D} = \{1, \ldots, n-1\}$. To all positions $p' \in \mathbb{D}^2$ not met by Π, we implicitly assume the blank shred n to be assigned. For such a solution a total error estimate is calculated as the sum of the errors imposed by all realized neighborhood relations of the shreds, including neighborhoods with the blank shred. Although at a first glance this representation might look rather unhandy, it turns out to be very useful, in particular since well matching sequences of shreds need not to be wrapped at the end of a row or column. Using efficient data structures and algorithms, the overhead of this solution representation is rather small (if not negligible). In situations where the exact dimensions of the original document(s) are known, the solution space can, of course, be defined smaller.

The remainder of the paper is organized as follows: In the next section a short overview on related work is given. In Sec. 3 we present the memetic algorithm with local improvement based on variable neighborhood search. Section 4 describes detailed experimental results comparing the proposed approaches with the best previously published method. Finally conclusions including an outlook on future research on this topic complete this article.

2 Related and Previous Work

The field of document reconstruction can be classified into various subdomains including among others the reconstruction of *strip shredded* (text) documents [21, 22], the restoration of *hand torn* paper documents [6, 11] and the

reconstruction of cross-cut shredded (text) documents (RCCSTD) [15, 16, 20]. Also, there is the large (sub-)field of restoration and reconstruction of historical fragmented documents and clay jugs as found in archeology, cf. [12]. Another related topic is the (computer aided) solving of jigsaw puzzles [4, 8].

Although these problems are similar on a first glance, they significantly differ in the details. For example, the pieces of a high quality jigsaw puzzle fit (almost) perfectly and uniquely into each other which obviously makes the result singular. Hand torn paper documents, on the other hand, might consist of snippets with unique shapes but due to frayed edges provoked by physical characteristics of paper, it is likely that two originally connected snippets will not perfectly fit together anymore. Finally, snippets produced during mechanically destroying multiple sheets of paper are in many cases shaped almost identical such that methods not only relying on the outer form of the pieces must be developed for correctly matching them.

Ukovich *et al.* [22], for instance, used MPEG-7 descriptors in the context of strip shredded documents. They extended their list of extracted features with characteristics especially related to text documents like line spacing or text color in [21]. In [16] Prandtstetter formulated this problem as a combinatorial optimization problem and showed that the reconstruction of strip shredded documents is $\mathcal{NP}$-complete. Since this problem represents a special case of RCCSTD, it can be concluded that RCCSTD is $\mathcal{NP}$-hard, too.

Other approaches published so far can be used as a preprocessing step for narrowing the search space; e.g., in [23] Ukovich *et al.* published a clustering algorithm to identify sets of strips most likely originating from the same input page. Since no relative order of the strips is produced an approach like the one published in [17] can be applied to gather the original document.

3 A Hybrid Approach for the Reconstruction of Cross-Cut Shredded (Text) Documents

For many real-world as well as academic optimization problems best results are frequently obtained by hybrid approaches, which combine different optimization strategies in order to exploit individual properties and benefit from synergy. Nowadays, a large variety of such hybridization techniques is known, from very simple, straight-forward methods to highly sophisticated and complex ones. General overviews on hybrid metaheuristics can, e.g., be found in [3, 19]. Here, we consider a memetic algorithm (MA) [14] which can be seen as a combination of a genetic algorithm (GA) [10] and a local improvement procedure. While the GA emphasizes diversification of the heuristic search, local improvement is supposed to "fine-tune" solutions, i.e., to focus on intensification. In our case, we realize the local improvement with a variable neighborhood search (VNS) [9] strategy. In addition to the intertwined execution of VNS as a subordinate of the MA, an extended version of the VNS is finally applied to the best solution obtained from the MA.

Algorithm 1: Memetic Algorithm for RCCSTD

```
begin
    t ← 0;
    Initialize the population P(t);
    repeat                                   // start the reproduction cycle
        t ← t + 1;
        Select(P(t − 1));                          // choose from the parents
        Recombine(P(t));            // generate & evaluate the descendants
        Mutate(P(t));                                 // mutate descendants
        Create new population P(t);     // new population from descendants
        ImproveSolutions(P(t));                 // improve some individuals
    until allowed generations reached;
    ImproveFinalSolution(P(t));                  // improve best individual
    return best individual;
```

The memetic algorithm's general framework is shown in Algorithm 1. In the following, we discuss its individual parts in detail.

3.1 Initial Population

For creating initial solutions we use two construction heuristics—namely the *row building heuristic* and the *Prim based heuristic*—originally introduced in [16], whereas 50% of the individuals of the first population were created using RBH and the other 50% by using PBH:

Row Building Heuristic (RBH): In general, text documents are laid out such that there is a blank margin on the left and the right of each page. Furthermore, it can be observed that usually shredded documents will not be exactly cut along the text line beginning, i.e., there will still be shreds with blank left or right margins. Obviously, it is very likely that these shreds should be placed in any final solution to the left and right edges of the reconstructed document page(s). Therefore, the row building heuristic starts to build rows by first randomly choosing a shred with a blank left margin. Subsequently, a best fitting shred—with respect to the error estimation function—is placed to the right of the already processed snippets. This process is repeated until a shred is added containing a blank right margin. Following the same procedure, a new line is started. If no more shreds with blank left margins are available, some other not yet processed shred is randomly chosen for starting the next row. This process continues until all shreds are placed.

Prim-Based Heuristic (PBH): Based on the same idea as the well-known algorithm of Prim [18] for determining minimum spanning trees, our Prim-based heuristic (PBH) starts with a randomly chosen shred. It then expands the partial solution iteratively by always adding the currently best matching shred to the according position, i.e., the solution grows from the initial shred

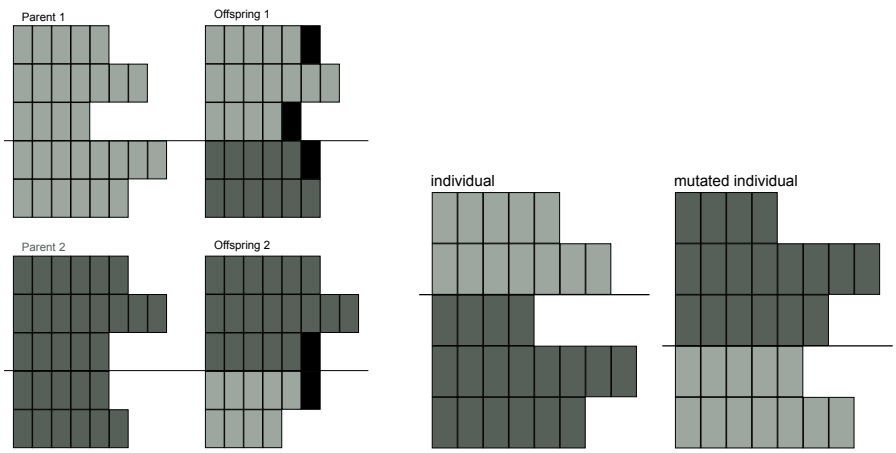


Fig. 1: Schematic representations of (a) the horizontal block crossover and (b) the horizontal flop mutation.

in general in all directions. Again, the process is iterated until all shreds are placed.

3.2 Recombination

Three special recombination operators, called *horizontal block crossover*, *vertical block crossover*, and *biased uniform crossover* are used in our MA. They are essentially two-dimensional extensions of standard crossover operators for strings.

Horizontal Block Crossover (HBX): This operator follows the idea of the *one-point crossover* introduced by Holland [5]. While in the latter a splitting *point* is randomly chosen, a splitting *line* needs to be computed in our two-dimensional case. To create one offspring all shreds on top of this line are taken from the first parent while all shreds below are adopted from the second parent, see Fig. 1a. A second descendant can be correspondingly obtained by taking the shreds on top of the splitting line from the second parent and the shreds below the line from the first parent. The position of the splitting line itself is randomly chosen based on a binomial distribution in the range $[1, r]$ with r indicating the minimum number of rows of the two parents.
Unfortunately, it might happen that after performing HBX shreds appear twice or not at all. To avoid the former case, all shreds already contained in the newly generated solution when adding the shreds below the splitting line are skipped. Those shreds missing at all are inserted into the offspring by using a best fit strategy that places them either at empty positions (of previously skipped shreds) or to the end of the rows, see black shreds in Fig. 1a.

Vertical Block Crossover (VBX): This operator forms the vertical equivalent to HBX, i.e., analogously to HBX, VBX randomly chooses a *vertical* splitting line and places all shreds left of this line according to the first parent and right of that line according to the second parent. A second offspring is derived correspondingly by switching the parents. Of course, it is again important to make sure that in the end each shred occurs exactly once.

Biased Uniform Crossover (BUX): This operator is as indicated by the name related to the standard uniform crossover on strings. Essentially analogously to it, for each position of the offspring either the shred of the first parent or the shred of the second parent is selected. BUX contains, however, some modifications to standard uniform crossover which became necessary primarily due to the uniqueness of the shreds and the two-dimensional problem representation.

Two offspring are again generated, and they inherit their shapes with respect to the positions where non-empty shreds are placed from the two parental solutions, respectively. Iteratively considering all the non-empty positions of the shape-defining parent, shreds are either adopted from the first or second parent. In contrast to standard uniform crossover, however, this decision is not made purely at random but using a best fit strategy, i.e., the shred is selected that fits better to the current position (with respect to the already placed shreds and the error estimation function). Furthermore, if one of the two candidate shreds is the virtual blank shred or has already been assigned to another position, then the other shred is chosen if not yet placed. If, both shreds have already been processed (or are blank), then a shifting in the assignment of shreds is performed, i.e., the shreds of the next position from the parents are considered for placing them at the offspring's current position. This procedure is iterated until all positions of the first parent are processed (or no more shreds to be assigned are left).

In the end, there might be some shreds which have not been assigned to the offspring. These missing shreds are finally placed in random order to the respectively best fitting positions (independent of the parental shape).

3.3 Mutation

In general mutation acts as diversification and innovation operator such that the algorithm does not converge too fast and new or lost genetic material is (re-)introduced. In our case, mutation also performs as regulator such that (negative) side-effects of the recombination operations can be compensated. Our MA includes the following four mutation operators:

Horizontal Flop (HFM): This operator is the mutating equivalent of HBX, i.e., the individual is horizontally split into two parts along a splitting line, which is randomly chosen based on a binomial distribution. The lower and upper parts of the parent are then swapped with each other, see Fig. 1b. Thus all left-right and top-bottom relations are retained except the relations along the splitting line.

Vertical Flop (VFM): This mutation complements HFM analogously to VBX, i.e., a vertical splitting line is chosen and the emerging left and right parts of the individuals are swapped. Since the lines of a solution in general do not share the same length, the left side of the descendant is filled with placeholders, i.e., blank shreds. Otherwise the new right half would be shifted to the end of each line and thus the top-bottom relations could not be conserved.

Break Line (BLM): This operator was designed because first tests showed that especially HBX and VBX create individuals whose lines become longer and longer over time. This effect is certainly evoked due to the best fit heuristic utilized that often adds remaining shreds to the end of lines. The idea is now to find the longest line of the parent individual and randomly, again using a binomial distribution, choose a point where this line is split. While the left part of the line remains where it is, the right part is added to the bottom of the document page.

Swap Two (S2M): Finally, S2M is the most simple but also most flexible mutation operator. The basic idea is to randomly swap two shreds. This operation is repeated up to ten times depending on a randomly chosen value.

3.4 Selection and Generation Replacement

The error estimation functions $c(i, j)$ and $\widehat{c}(i, j)$ from [16] perform well for our purpose as a fitness function but cannot garantee that the error induced by the original document is the minimal error, i.e., there might exist a shred assigment with an overall error smaller than the error of the correctly reconstructed document. These "better" solutions are represented by a negative value in Tab. 1. At this point it should also be mentioned that individuals are seen as fitter than others if the estimated total error of the corresponding solution is smaller. Because of this problem a fitness proportional selection depending on these error estimation functions does not seem adequate.

Therefore in preliminary experiments the following selection and generation replacement scheme, which is somehow related to the $(\mu + \lambda)$ model of evolution strategies [2], turned out to be superior to a classical fitness-proportional scheme: A new generation is built by first copying the 10% best individuals (elitists) of the previous population. The remaining 90% are filled with newly created individuals, which are derived by recombining uniformly selected parents and performing mutation. Because of the crossover operators' computational complexity only as many descendants are created as necessary to fill the next population. From the two offspring each call of HBX or VBX yields, only the better one is kept.

3.5 Local Improvement by Variable Neighborhood Search

The MA includes a VNS as embedded procedure for improving candidate solutions and in particular also the final solution. More precisely, this VNS is a so-called *general VNS*, which itself further includes a *variable neighborhood descent* (VND) procedure, i.e., a systematic local search utilizing multiple neighborhood

structures [9]. The following paragraphs detail the VND/VNS framework and their application within the MA.

Variable Neighborhood Descent (VND)

Based on the concept of local search, VND tries to systematically examine predefined neighborhood structures $\mathcal{N}_l$, with $1 \leq l \leq l_{\max}$, to transform a given initial solution into one that is a local optimum w.r.t. all these neighborhood structures. Usually, the neighborhoods are ordered such that the smaller ones are searched first and the larger ones are only examined if no further improvements in the smaller ones can be achieved. For this work we adopted the moves and neighborhood structures originally presented in [15]:

SwapMove(i, j) swaps two shreds i and j, with $i, j \in S$.

ShiftMove(p, w, h, d, a) moves a rectangular region of snippets, whereas parameter $p = (x, y) \in \mathbb{D}^2$ indicates the top left shred of this rectangle, the length and width of the rectangle are given by $w \geq 1$ and $h \geq 1$, parameter $a \geq 1$ indicates the amount of shreds the rectangle should be shifted, and d specifies the direction, i.e., horizontally (left or right) or vertically (up or down). Previously adjacent shreds are suitably shifted.

Based on these two move types the following neighborhoods are defined:

$\mathcal{N}_1$ All solutions reachable by a single swap move.

$\mathcal{N}_2$ All solutions reachable by a single shift move of one single shred in either x or y direction.

$\mathcal{N}_3$ All solutions obtained by applying a single shift move with either parameter w or parameter h set to one, while the other parameters can be chosen freely.

$\mathcal{N}_4$ As neighborhood $\mathcal{N}_3$ but all parameters can be freely chosen.

$\mathcal{N}_5$ All solutions reachable by two consecutive shift moves applied to a single shred, whereas the first shift move moves the shred along the x-axis and the second one along the y-axis.

$\mathcal{N}_6$ Analogously to $\mathcal{N}_5$ this neighborhood contains all solutions reachable by two consecutive shift moves (first along x-axis, second along y-axis), but instead of displacing a single shred a larger rectangle of either width one or height one is considered.

$\mathcal{N}_7$ This neighborhood is the most general one, induced by two consecutive shift moves (first along x-axis, second along y-axis) of an arbitrarily sized rectangle, i.e., p, w, h and a are chosen freely.

Obviously, an implicit order of the neighborhoods is given since $\mathcal{N}_i$ contains $\mathcal{N}_{i-1}$ for $i = 3, 4, 6, 7$, i.e., the number of candidate solutions within $\mathcal{N}_i(\Pi)$ will in general be larger than the number of candidates in $\mathcal{N}_{i-1}(\Pi)$.

To achieve a reasonable runtime of this VND, an efficient implementation is crucial. Therefore, an incremental update function for estimating the error is utilized. In addition, we assume that at least one position $p \in \mathbb{D}^2$ is affected by each move since the shifting/swapping of empty shreds has either a negative effect on the error made or none at all. Finally, we use a next improvement strategy to further reduce the overall running time.

Variable Neighborhood Search (VNS)

The VND described in the last paragraphs in general only yields solutions that are locally optimal with respect to the considered neighborhood structures. To further enlarge the improvement potential, VND is embedded in a variable neighborhood search (VNS) that performs shaking on the basis of additional larger neighborhood structures. These additional neighborhood structures are defined by a series of shift moves of single shreds, where in VNS neighborhood structure N_i, with $1 \leq i \leq 5$, i^2 randomly chosen shift moves are applied.

Embedding of the VNS in the MA

In the MA's inner local improvement phase, only a reduced variant of the above described VNS is applied due to runtime reasons. This reduced variant only considers the three smallest neighborhood structures, i.e., $\mathcal{N}_1$, $\mathcal{N}_2$ and $\mathcal{N}_3$ within the VND. Furthermore, this local improvement is only executed every 5000 generations and only to the best 10% of the individuals of the current population.

The best solution of the MA's last generation is finally improved in a more aggressive way by applying the full VNS, i.e., utilizing also the more complex neighborhood structures $\mathcal{N}_4$ to $\mathcal{N}_7$ within the VND.

4 Experimental Results

Within this section computational results are presented including a description of the experimental setups and the used benchmark instances.

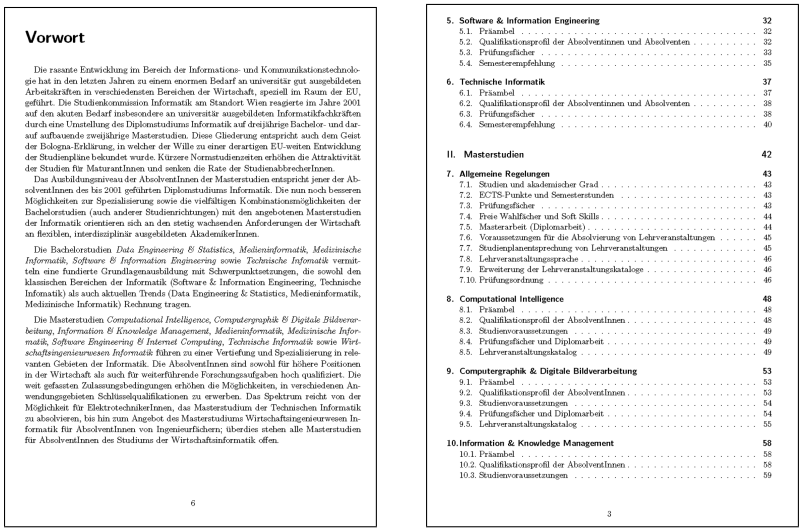
4.1 Test Instances

As benchmark instances we used the same as in [15]. These instances were generated by virtually cutting five text document pages using nine different cutting patterns ranging from 9×9 to 15×15 shreds. Moreover, all pages were scanned with a resolution of 1240×1755. According to our definition of RCCSTD, all blank shreds are replaced by the single virtual shred. The instances p01 and p02 can be seen in Fig. 2, while a visual representation of the complete set can be found in [16].

4.2 Experimental Setups

For each of the test series, the population size was set to 300 individuals and the number of generations was 30000. Based on preliminary tests we selected the following combinations of crossover operators and mutation operators to be examined in detail:

HBX+VBX It turned out that HBX and VBX perfectly complement each other such that this test setting applies to 50% of all pairs of parents HBX and to the other 50% VBX, whereas 270 pairs of parents are randomly



Vorwort

Die rasante Entwicklung im Bereich der Informations- und Kommunikationstechnologie hat in den letzten Jahren zu einem enormen Bedarf an universitär gut ausgebildeten Arbeitskräften in verschiedensten Bereichen der Wirtschaft, speziell im Raum der EU, geführt. Die Studienkommission Informatik am Standort Wien reagierte im Jahre 2001 auf den akuten Bedarf insbesondere an universitär ausgebildeten Informatikfachkräften durch eine Umstellung des Diplomstudiums Informatik auf dreijähriges Bachelor- und darauf aufbauende zweijährige Masterstudien. Diese Gliederung entspricht auch dem Geist der Bologna-Erklärung, in welcher der Wille zu einer derartigen EU-weiten Entwicklung der Studienpläne bekundet wurde. Kürzere Normstudienzeiten erhöhen die Attraktivität der Studien für MaturantInnen und senken die Rate der StudienabbrecherInnen.

Das Ausbildungsniveau der AbsolventInnen der Masterstudien entspricht jener der AbsolventInnen des bis 2001 geführten Diplomstudiums Informatik. Die nun noch besseren Möglichkeiten zur Spezialisierung sowie die vielfältigen Kombinationsmöglichkeiten der Bachelorstudien (auch anderer Studienrichtungen) mit den angebotenen Masterstudien der Informatik orientieren sich an den stetig wachsenden Anforderungen der Wirtschaft an flexiblen, interdisziplinär ausgebildeten AkademikerInnen.

Die Bachelorstudien *Data Engineering & Statistics, Medieninformatik, Medizinische Informatik, Software & Information Engineering* sowie *Technische Informatik* vermitteln eine fundierte Grundlagenausbildung mit Schwerpunktsetzungen, die sowohl den klassischen Bereichen der Informatik (Software & Information Engineering, Technische Informatik) als auch aktuellen Trends (Data Engineering & Statistics, Medieninformatik, Medizinische Informatik) Rechnung tragen.

Die Masterstudien *Computational Intelligence, Computergraphik & Digitale Bildverarbeitung, Information & Knowledge Management, Medieninformatik, Medizinische Informatik, Software Engineering & Internet Computing, Technische Informatik* sowie *Wirtschaftsingenieurwesen Informatik* führen zu einer Vertiefung und Spezialisierung in relevanten Gebieten der Informatik. Die AbsolventInnen sind sowohl für höhere Positionen in der Wirtschaft als auch für weiterführende Forschungsaufgaben hoch qualifiziert. Die weit gefassten Zulassungsbedingungen erhöhen die Möglichkeiten, in verschiedenen Anwendungsgebieten Schlüsselqualifikationen zu erwerben. Das Spektrum reicht von der Möglichkeit für Elektrotechnikerinnen, das Masterstudium der Technischen Informatik zu absolvieren, bis hin zum Angebot des Masterstudiums Wirtschaftsingenieurwesen Informatik für AbsolventInnen von Ingenieurfächern; überdies stehen alle Masterstudien für AbsolventInnen des Studiums der Wirtschaftsinformatik offen.

6

(a)

3

(b)

Fig. 2: (a) instance p01 and (b) instance p02

selected in each generation. For each crossover operator only the better of the two obtained descendants is used, while the other is discarded. This results in 270 newly created descendant while the remaining 30 individuals are the adopted elite of the previous generation. To 25% of all individuals mutation was applied whereas the concrete probabilities for choosing each mutation operator are the following: 5% HFM, 5% VFM, 10% BLM and 5% S2M.

BUX With this configuration the *biased uniform crossover* is tested. Since no other crossover operator is used in this setup both descendants of the operator were used for the next population. Again, 25% of the newly created individuals are mutated, but here the individual probabilities are 5% HFM, 15% VFM and 5% S2M.

For both basic setups we wanted to investigate in particular the influence of the local search phase(s), i.e., a pure genetic algorithm, a memetic algorithm, and the memetic algorithm followed by a more sophisticated VNS are compared. This leads in total to six different settings.

4.3 Test Results

Table 1 shows the results of the six tested configurations together with the so far best results obtained by the ACO from [15]. As a local improvement this ACO incorporates the same VND as described above using the first three neighborhood structures, $\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3$, as the MA does in its inner local improvement

Table 1: The mean percentage gaps of objective values over 30 runs and standard deviations for the tested configurations. For comparison the ACO results from [15] are also presented. The symbols in columns p_{ACO/BV^2}, p_{ACO/HV^2}, p_{BV^2/HV^2} indicate the statistical significance of differences according to student t-test with an error level of 1%.

			ACO	B	BV	p_{ACO/BV^2}	BV^2	p_{BV^2/HV^2}	H	HV	p_{ACO/HV^2}	HV^2
	x	y/ orig	gap/dev [%]/ [%]	gap/ dev [%]/ [%]	gap/dev [%]/ [%]		gap/dev [%]/ [%]		gap/ dev [%]/ [%]	gap/ dev [%]/ [%]		gap/ dev [%]/ [%]
instance p01	9	9/2094	3,7/ 7,2	0,0/ 0,0	0,0/ 0,0	>	0,0/ 0,0	≈	0,0/ 0,0	0,0/ 0,0	>	0,0/ 0,0
	9	12/3142	30,4/ 4,2	39,3/ 6,9	12,2/ 4,4	>	11,9/ 4,2	≈	20,4/ 8,8	9,6/ 5,3	>	9,3/ 5,4
	9	15/3223	33,9/ 3,9	45,8/ 8,5	9,8/ 5,4	>	9,1/ 5,7	≈	25,5/14,0	9,2/ 6,9	>	9,1/ 6,7
	12	9/2907	32,1/ 5,2	44,1/ 8,8	13,2/ 4,7	>	12,6/ 4,8	>	18,6/12,6	7,6/ 9,1	>	7,6/ 9,0
	12	12/3695	31,3/ 3,7	43,7/ 6,5	8,9/ 2,5	>	8,4/ 2,6	≈	28,2/ 4,0	8,5/ 2,7	>	8,3/ 2,8
	12	15/3825	36,0/ 2,8	54,8/ 6,1	11,8/ 1,9	>	11,4/ 2,0	≈	34,7/ 6,0	10,6/ 2,1	>	10,5/ 2,2
	15	9/2931	39,2/ 5,2	39,6/ 6,4	10,2/ 6,5	>	10,0/ 6,3	>	1,9/ 5,9	2,0/ 6,4	>	1,9/ 6,3
	15	12/3732	34,5/ 2,3	48,1/ 5,6	12,7/ 3,4	>	12,5/ 3,5	≈	27,1/ 8,6	10,0/ 4,4	>	9,9/ 4,3
	15	15/3870	39,2/ 2,8	52,4/ 7,1	15,6/ 2,2	>	15,2/ 2,3	>	39,3/ 5,4	11,7/ 3,0	>	11,4/ 2,9
instance p02	9	9/1434	-3,8/ 5,1	0,3/11,6	-28,4/ 1,6	>	-28,6/ 1,6	≈	-24,0/ 4,6	-29,1/ 1,4	>	-29,2/ 1,3
	9	12/1060	23,6/ 5,5	63,3/21,9	2,9/ 1,8	>	2,7/ 1,7	>	9,1/ 7,8	0,9/ 1,7	>	0,9/ 1,7
	9	15/1978	7,9/ 2,8	29,0/10,5	-8,4/ 1,0	>	-9,0/ 0,9	≈	1,8/ 5,4	-9,4/ 1,9	>	-9,5/ 1,9
	12	9/1396	6,4/ 4,8	-1,6/11,6	-26,8/ 2,4	>	-27,6/ 2,0	>	-15,2/ 8,7	-28,9/ 2,3	>	-29,2/ 2,1
	12	12/1083	31,6/ 5,7	34,4/13,3	1,5/ 2,7	>	1,3/ 2,6	≈	8,9/11,2	0,3/ 2,1	>	0,2/ 1,9
	12	15/1904	12,4/ 3,4	14,5/11,4	-9,7/ 1,7	>	-10,1/ 1,6	≈	1,0/ 7,6	-9,3/ 2,2	>	-9,3/ 2,2
	15	9/1658	10,7/ 4,4	12,7/ 7,5	-11,6/ 2,3	>	-12,0/ 2,1	>	-7,1/ 7,8	-14,3/ 2,9	>	-14,3/ 2,9
	15	12/1503	17,5/ 4,8	21,5/ 8,0	1,9/ 2,2	>	1,8/ 2,2	>	3,7/ 6,0	0,4/ 1,3	>	0,4/ 1,3
	15	15/2283	11,9/ 2,4	13,0/ 7,7	-5,0/ 1,7	>	-5,2/ 1,7	≈	5,6/ 7,1	-5,1/ 2,5	>	-5,2/ 2,5
instance p03	9	9/2486	10,0/ 5,6	4,8/ 8,3	-7,6/ 1,7	>	-7,6/ 1,7	≈	-4,0/ 7,0	-5,6/ 5,0	>	-5,6/ 5,0
	9	12/2651	35,0/ 3,6	32,2/ 9,1	6,7/ 3,6	>	6,3/ 3,7	≈	13,3/12,3	4,7/ 6,5	>	4,3/ 6,1
	9	15/2551	23,0/ 3,9	26,6/ 7,4	2,0/ 2,9	>	1,8/ 2,9	>	2,8/ 5,8	-0,1/ 1,4	>	-0,1/ 1,4
	12	9/3075	15,4/ 3,0	18,7/ 3,9	5,8/ 2,4	>	5,6/ 2,4	≈	11,4/ 4,8	4,7/ 2,2	>	4,7/ 2,2
	12	12/3377	26,2/ 3,2	33,5/ 6,0	3,6/ 2,5	>	3,0/ 2,5	≈	18,9/ 7,4	1,8/ 4,4	>	1,8/ 4,4
	12	15/3313	18,1/ 2,0	26,9/ 5,8	2,1/ 2,7	>	1,7/ 2,8	>	4,0/ 6,5	-2,9/ 1,1	>	-3,0/ 1,0
	15	9/3213	19,4/ 3,0	27,9/ 6,3	-0,7/ 4,0	>	-1,0/ 3,9	≈	11,0/ 7,2	0,2/ 3,0	>	0,1/ 3,0
	15	12/3278	41,6/ 3,8	51,7/ 6,2	10,5/ 4,1	>	10,1/ 4,1	≈	27,4/ 9,1	12,6/ 5,2	>	12,3/ 5,1
	15	15/3308	26,4/ 2,8	44,8/ 5,2	5,8/ 2,4	>	5,4/ 2,4	>	6,8/ 7,5	0,8/ 2,0	>	0,7/ 2,0
instance p04	9	9/1104	22,9/ 7,8	19,6/12,9	-27,1/ 2,9	>	-28,0/ 2,7	<	-15,3/13,7	-20,6/11,9	>	-20,9/11,9
	9	12/1463	11,6/ 4,7	15,6/ 8,5	-13,1/ 4,9	>	-13,7/ 5,0	≈	-3,8/10,2	-12,8/ 4,7	>	-13,1/ 4,6
	9	15/1589	-0,3/ 4,0	-0,2/ 5,9	-20,7/ 3,4	>	-20,8/ 3,4	≈	-17,1/ 6,9	-22,2/ 2,9	>	-22,3/ 2,9
	12	9/1515	34,7/ 6,3	38,8/ 9,2	-8,1/ 6,4	>	-8,3/ 6,3	<	16,2/17,4	-3,3/ 5,7	>	-3,4/ 5,7
	12	12/2051	17,8/ 3,2	19,8/ 5,3	3,2/ 2,7	>	3,0/ 2,6	>	13,2/ 6,4	-3,7/ 3,4	>	-3,9/ 3,4
	12	15/2146	4,0/ 2,6	3,9/ 4,7	-11,9/ 2,4	>	-12,4/ 2,4	>	-10,6/ 5,7	-18,9/ 1,5	>	-19,0/ 1,6
	15	9/1567	17,8/ 5,5	26,1/11,3	-0,5/ 5,5	>	-1,0/ 5,6	≈	11,5/ 9,9	0,9/ 3,8	>	0,9/ 3,8
	15	12/1752	33,6/ 4,8	32,3/ 9,6	8,9/ 3,5	>	8,5/ 3,4	≈	24,9/ 9,9	6,1/ 4,5	>	6,0/ 4,5
	15	15/2026	2,8/ 2,8	10,8/ 6,2	-8,8/ 1,6	>	-8,9/ 1,6	≈	-3,2/ 5,9	-9,1/ 2,3	>	-9,1/ 2,2
instance p05	9	9/ 690	19,0/ 9,1	0,0/ 0,0	0,0/ 0,0	>	0,0/ 0,0	≈	0,0/ 0,0	0,0/ 0,0	>	0,0/ 0,0
	9	12/ 888	86,6/ 7,4	79,7/19,0	8,0/ 8,8	>	7,0/ 8,4	≈	30,3/26,6	7,8/10,6	>	7,6/10,4
	9	15/1623	43,1/ 4,4	57,3/ 7,8	9,9/ 4,8	>	9,0/ 4,7	≈	28,6/14,1	7,4/ 4,8	>	7,2/ 4,6
	12	9/1016	31,0/ 4,2	15,6/12,3	0,7/ 3,1	>	0,6/ 2,9	≈	2,8/ 7,4	-0,1/ 2,2	>	-0,1/ 2,2
	12	12/1325	41,5/ 5,5	50,9/16,7	2,0/ 5,5	>	1,4/ 5,6	>	17,4/18,4	-7,5/ 7,4	>	-8,1/ 6,2
	12	15/1986	39,6/ 3,6	61,9/ 9,6	16,0/ 3,1	>	15,3/ 3,0	>	35,7/12,4	9,3/ 4,5	>	9,0/ 4,5
	15	9/1010	-9,6/ 4,0	-14,8/ 9,9	-18,8/ 2,1	>	-18,8/ 2,1	≈	-18,8/ 2,1	-19,3/ 0,5	>	-19,3/ 0,5
	15	12/1156	57,8/ 8,8	66,1/19,4	5,4/ 7,3	>	4,8/ 7,4	>	14,9/24,4	-6,4/ 8,3	>	-6,6/ 8,0
	15	15/1900	36,3/ 3,6	73,0/10,7	5,1/ 4,1	>	4,2/ 3,8	>	41,9/ 8,4	1,7/ 4,3	>	1,0/ 4,0

phase. During the ACO this local improvement is applied to each solution every iteration. Because in [15] is shown that the ACO clearly outperforms the VNS alone, the results of the MA are only compared to the ACO within this work.

The labels of the columns should be interpreted as follows: H, HV, HV^2 indicate the three test settings based on the combination of HBX and VBX solely (genetic algorithm), with intertwined VNS (memetic algorithm) and with intertwined VNS and VNS at the end (extended hybrid approach). Analogously, B, BV, BV^2 refer to the results obtained by a pure genetic algorithm utilizing BUX, the corresponding memetic algorithm (BV) and the hybrid of the memetic algorithm and the VNS (BV^2). The column labeled ACO refers to the results obtained by the ACO. For each setting the mean percentage gaps over 30 runs are presented together with the standard deviations (columns gap and dev), whereas the percentage gap indicates the relative gap to the calculated estimated error of the original document (shown in the column labeled with orig). The first two columns list the applied cutting patterns for each instance (shreds along the x-axis and along the y-axis). Finally, the columns labeled p_{ACO/BV^2}, p_{ACO/HV^2} and p_{BV^2/HV^2} indicate the statistical significance of differences according to student t-test with an error level of 1%: A "$<$" means that algorithm A is significantly better than algorithm B, where algorithm A is the ACO in case of column $p^2_{ACO/BV}$ and $p^2_{ACO/HV}$ and BV^2 for p_{BV^2/HV^2}. Algorithm B is HV^2 for p_{ACO/HV^2} and p_{BV^2/HV^2} and BV^2 for p_{ACO/BV^2}. A "$>$" indicates that algorithm B is significantly better than algorithm A and a "$\approx$" shows that no significant difference can be observed for that instance.

However, the following trend can be extended from the results in Tab. 1: related to the GA alone the settings based on HBX and VBX are far better than the settings based on BUX, whereas the intertwined VNS slightly compensates the positive effects of HBX and VBX. The performance of the MAs is, however, clearly better than the performance of the pure GAs.

Furthermore, even for BV^2 and HV^2 a trend to HV^2 can be detected as indicated by column p_{BV^2/HV^2}. As verified by columns p_{ACO/BV^2} and p_{ACO/HV^2} both settings are clearly better than the so far best known approach. (Even the settings H, HV and BV are better than the ACO based approach—not explicitly shown in the table).

All tests were executed with 3GB of RAM on a single core of an Intel Xeon (Nehalem) Quadcore CPU with 2.53 GHz. The computation times for ACO lie between 10 seconds and 800 seconds. The B setup needed between 100 seconds and 400 seconds, while H lies between 200 seconds and 700 seconds. Incorporating the VNS for BV^2 the computational times vary between 150 seconds to 4000 seconds and for HV^2 from 200 seconds to 2500 seconds.

4.4 Reconstruction Results

In Fig. 3 four reconstruction possibilities of the instances p01 and p02 are shown, whereas the virtual shreds are not printed and thus the number of shreds shown is smaller than the original. All shreds are seperated by gray edges. A light grey

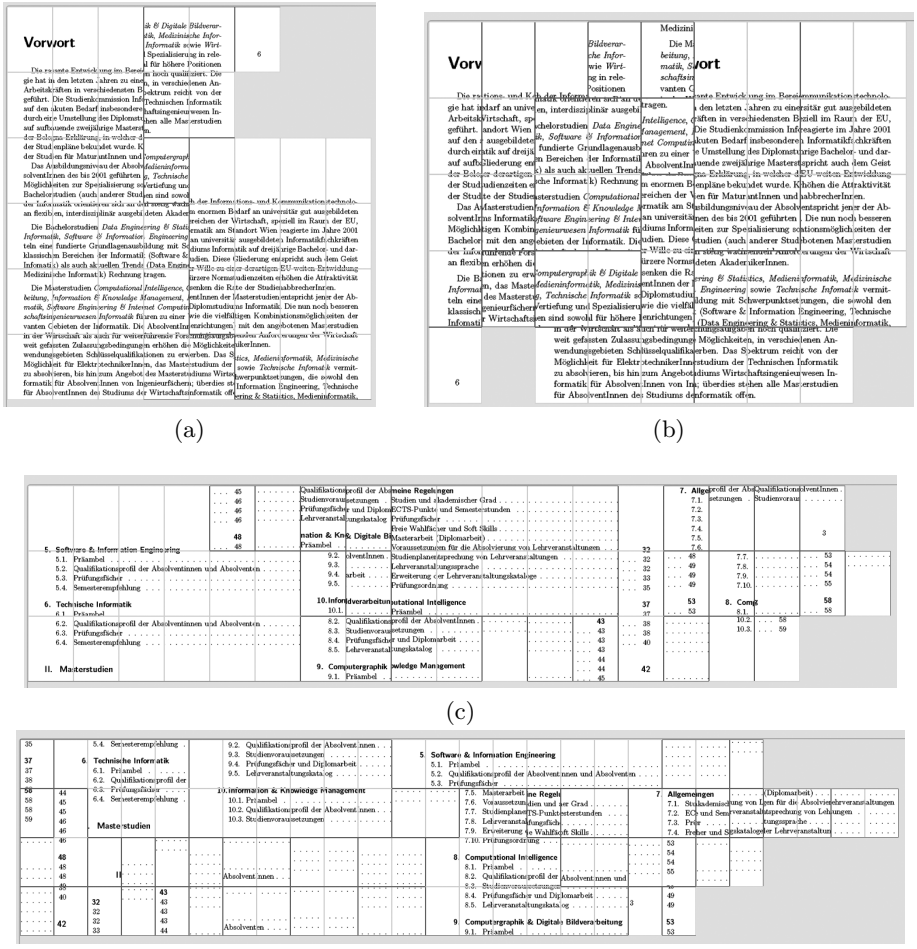


Fig. 3: (a) and (b) show reconstructions of p01, (c) and (d) of p02

edge indicates that along this border the two shreds are positioned correctly, while dark grey symbolizes an incorrect assignment.

Fig. 3a shows instance p01 with a cutting pattern of 9×9 and was reconstructed with a percentage gap of 6.1% due to original. Note that great parts are reconstructed correctly beside the wrong position of two right blocks, which leads to the dark gray line seperating the left and right blocks. Fig. 3b shows another reconstruction of p01 with a percentage gap of 16.3%.

Two possible reconstructions of instance p02 are shown in Fig. 3c cut 9×9 with a percentage gap of -1% and Fig. 3d cut 12×12 with -1.7%. Because of

many shreds that contain little information a reconstruction with an overall error smaller than the error indicated by the original document is possible. Therefore great parts of the text passages could be reconstructed, while the correct position of these blocks could not be determined.

5 Conclusions and Future Work

In this work we presented a memetic algorithm (MA) to reconstruct cross-cut shredded text documents. This hybrid approach is based on a genetic algorithm (GA) extended with a local search procedure, which is realized in form of a variable neighborhood search (VNS) incorporating a variable neighborhood descent (VND). In addition, we presented a hybrid algorithm combining the MA with a more sophisticated version of the VNS. For the GA/MA three different crossover and four mutation operators were designed and implemented. The MA was then tested on five document pages shredded into nine different cutting patterns each, which leads to 45 different test instances. Based on these test instances we compared the algorithms presented within this paper with each other as well as with the so far best known approach which is based on an ant colony optimization method.

The results obtained suggest that the proposed GA/MA mainly based on a two-dimensional version of a one-point crossover clearly outperforms the ACO approach. Even more, the hybrid MA/VNS version, i.e., the subsequently executed VNS, could further improve the results obtained by the pure MA.

The results indicate, however, that for relatively small instances it is possible to clearly reconstruct the original documents. Nevertheless, future research in this area is necessary, e.g., it would be of great interest to develop crossover operators respecting the relative positions of shreds to each other. Moreover, other (metaheuristic) approaches should be followed to further improve the obtained results. Finally, some work needs to be done for further developing the error estimation function such that the results become more reliable.

References

1. Ávila, B.T., Lins, R.D.: A fast orientation and skew detection algorithm for monochromatic document images. In: DocEng '05: Proceedings of the 2005 ACM symposium on Document Engineering. pp. 118–126. ACM, New York, USA (2005)
2. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press (1996)
3. Blum, C., Augilera, M.J.B., Roli, A., Sampels, M. (eds.): Hybrid Metaheuristics – An Emergent Approach for Combinatorial Optimization, Studies in Computational Intelligence, vol. 114. Springer (2008)
4. Chung, M.G., Fleck, M.M., Forsyth, D.A.: Jigsaw puzzle solver using shape and color. In: Fourth International Conference on Signal Processing Proceedings 1998. Signal Processing Proceedings, vol. 2, pp. 877–880 (1998)
5. Davis, L. (ed.): Handbook of genetic algorithms. International Thomson Publishing Services, 1st edn. (1996)

6. De Smet, P.: Reconstruction of ripped-up documents using fragment stack analysis procedures. Forensic science international 176(2), 124–136 (2008)
7. Glover, F.W., Kochenberger, G.A. (eds.): Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 57. Kluwer Academic Publishers, New York, USA (2003)
8. Goldberg, D., Malon, C., Bern, M.: A global approach to automatic solution of jigsaw puzzles. Computational Geometry 28(2–3), 165–174 (2004)
9. Hansen, P., Mladenović, N.: Variable neighborhood search. In: Glover and Kochenberger [7], pp. 145–184
10. Holland, J.: Adaptation In Natural and Artificial Systems. University of Michigan Press (1975)
11. Justino, E., Oliveira, L.S., Freitas, C.: Reconstructing shredded documents through feature matching. Forensic Science International 160(2–3), 140–147 (July 2006)
12. Kleber, F., Diem, M., Sablatnig, R.: Torn document analysis as a prerequisite for reconstruction. In: Sablatnig, R., et al. (eds.) 15th International Conference on Virtual Systems and Multimedia. pp. 143–148. IEEE (2009)
13. Lu, S., Tan, C.L.: Automatic detection of document script and orientation. In: International Conference on Document Analysis and Recognition – ICDAR 2007. vol. 1, pp. 237–241. IEEE Computer Society, Los Alamitos, CA, USA (2007)
14. Moscato, P., Cotta, C.: A gentle introduction to memetic algorithms. In: Glover and Kochenberger [7], pp. 105–144
15. Prandtstetter, M., Raidl, G.R.: Meta-heuristics for reconstructing cross cut shredded text documents. In: Raidl, G.R., et al. (eds.) GECCO '09: Proceedings of the 11th annual conference on Genetic and evolutionary computation. pp. 349–356. ACM Press (2009)
16. Prandtstetter, M.: Hybrid Optimization Methods for Warehouse Logistics and the Reconstruction of Destroyed Paper Documents. Ph.D. thesis, Vienna University of Technology (2009)
17. Prandtstetter, M., Raidl, G.R.: Combining forces to reconstruct strip shredded text documents. In: Blesa, M., et al. (eds.) Hybrid Metaheuristics. LNCS, vol. 5296, pp. 175–189. Springer (2008)
18. Prim, R.C.: Shortest connection networks and some generalizations. The Bell System Technical Journal 3, 1389–1401 (1957)
19. Raidl, G.R., Puchinger, J., Blum, C.: Metaheuristic hybrids. In: Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics. Springer, 2nd edn. (accepted 2009, to appear)
20. Schauer, C.: Reconstructing Cross-Cut Shredded Documents by means of Evolutionary Algorithms. Master's thesis, Vienna University of Technology, Institute of Computer Graphics and Algorithms (2010)
21. Ukovich, A., Ramponi, G.: Features for the reconstruction of shredded notebook paper. IEEE International Conference on Image Processing, 2005. 3, 93–96 (2005)
22. Ukovich, A., Ramponi, G., Doulaverakis, H., Kompatsiaris, Y., Strintzis, M.: Shredded document reconstruction using MPEG-7 standard descriptors. Proceedings of the Fourth IEEE International Symposium on Signal Processing and Information Technology, 2004. pp. 334–337 (2004)
23. Ukovich, A., Zacchigna, A., Ramponi, G., Schoier, G.: Using clustering for document reconstruction. In: Dougherty, E.R., et al. (eds.) Image Processing: Algorithms and Systems, Neural Networks, and Machine Learning. Proceedings of SPIE, vol. 6064, pp. 168–179. International Society for Optical Engineering (2006)