Pack as pack can

Routinemäßige Messungen durch Automatik ersetzt. Packprogramme im Vergleich.

Franz Fiala DSK-478, 498

Immer wieder gibt es Leistungsvergleiche von Komprimierprogrammen, zuletzt im PCC-Magazin-29. Die PCNEWS edit wollen dem nicht nachstehen und die Vergleichsmöglichkeiten durch eine Besonderheit ergänzen.

Wozu packen?

Packen, d.h. Komprimieren oder Verkleinern von Dateien durch Entfernung redundanter Informationsanteile wird vielfältig eingesetzt. Es spart Speicherplatz, es spart Übertragungszeit, es vereinfacht die Archivierung zusammengehöriger Dateien.

Komprimieren von Daten ist schon so selbstverständlich geworden, daß es schon zum Hilfsmittel des Betriebssystems wurde (Stacker, DoubleDos) und ein eigenes Modemprotokoll (V.42). Die Bedeutung der Packer auf einer Festplatte ist daher geringer geworden, da eine zweifache Komprimierung keine weiteren Platzvorteile verschafft.

Aber es ist nach wie vor vorteilhaft, Programme und zusammegehörige Dateien in gepackten Zustand weiterzugeben, da nur auf diese Weise die Datenintegrität gewahrt bleibt. Es handelt sich also auch um eine Archiviermethode, die an die Fähigkeiten eines Backup-Programms heranreicht.

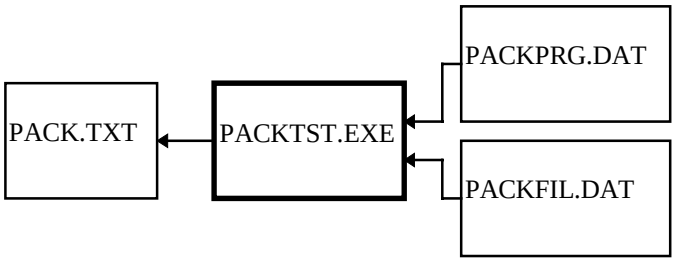
Alle Dateien der PCNEWS edit wurden vor einiger Zeit einheitlich ins ZIP-Format gebracht; warum gerade ZIP, erfahren Sie in diesem Beitrag.

Wie wurde getestet?

Üblicherweise wird man Programme vergleichen, indem man sie mit geeigneten Testdateien aufruft, die Pack- und Entpackzeiten und das Packverhältnis in Tabellenform zusammenfaßt. Mehr oder weniger Handarbeit.

Jede Veränderung der Testbedingungen erfordert, daß alles oder zumindest vieles wiederholt werden muß. Eine Automatik in Form eines Testprogramms mußte her. Alle nachfolgend abgedruckten Daten werden durch das Programm `PACKTST.EXE` generiert.

Die Parameter werden dem Programm über Listen mitgeteilt, das Programm arbeitet den Auftrag der Listen ab. Die variablen Daten der Packer werden über die Datei `PACKPRG.DAT` und jene der Testdateien über `PACKFIL.DAT` eingelesen. Damit kann jeder seine eigenen Programme vergleichen, wenn es neuere Versionen gibt oder wenn andere Optionen verglichen werden sollen.



Voraussetzungen

Der Aufbau der beiden Datendateien Datei `PACKPRG.DAT` und `PACKFIL.DAT` befolgt folgende Regeln:

- Alle erforderlichen Felder müssen vollständig angegeben sein
- Die Einträge sind durch SPACE oder TAB getrennt
- In den einzelnen Bezeichnern kommt kein SPACE oder TAB vor; sollte es erforderlich sein, ist das Feld durch ".." einzuschließen
- Leere Felder sind als "" oder ‘’ einzugeben
- Strichpunkte am Zeilenanfang machen die jeweilige Zeile unwirksam

Packprogrammliste PACKPRG.DAT

Der Aufbau der Datei `PACKPRG.DAT` ist wie folgt:

```
<Laufwerk>
<Pfad>
{
<name> <packprogramm> <entpackprogramm> <packoptionen> <entpackoptionen>
<extension> <version> <mode>
}
```

wobei

`<Laufwerk>`	Laufwerk auf dem sich die Packer befinden. "", wenn aktuelles Laufwerk
`<Pfad>`	Pfad (ohne schließenden Backslash) in dem sich die Packer befinden, "", wenn im aktuellen Pfad oder im Suchpfad zu finden
`{}`	wiederholen für alle Packprogramme
`<name>`	Kennzeichnung des Programms (erscheint in der ersten Spalte der Ergebnisdatei)
`<packprogramm>`	Programmname des Packprogramms ohne Erweiterung
`<entpackprogramm>`	Programmname des Entpackprogramms ohne Erweiterung
`<packoptionen>`	Optionen des Packprogramms
`<entpackoptionen>`	Optionen des Entpackprogramms
`<extension>`	Erweiterung durch den Packer
`<version>`	Version des Packprogramms (erscheint in der zweiten Spalte der Ergebnisdatei)
`<mode>`	0..normal 1..Kein eigener Packdateiname im Kommandostring

Beispiel

```
D:
\pcn\4x\43\zip
GZIP GZIP GZIP -SZZZ "-SZZZ -d" ZZZ 1.1.1 1
PKZIP/PKUNZIP PKZIP PKUNZIP "" "" ZIP 2.04g 0
LHA LHA LHA A X LZH 2.13 0
ZOO ZOO ZOO -add -extract ZOO 2.1 0
ARJ ARJ ARJ A X ARJ 2.41a 0
PKARC/PKXARC PKARC PKXARC A -E ARC 3.5 0
```

Vergleichsdateienliste PACKFIL.DAT

Der Aufbau der Datei `PACKFIL.DAT` ist wie folgt:

```
<Laufwerk>
<Pfad>
{
<name> <bezeichnung> <extension>
}
```

wobei

`<Laufwerk>`	Laufwerk auf dem sich die Testdateien befinden. "", wenn aktuelles Laufwerk
`<Pfad>`	Pfad (ohne schließenden Backslash) in dem sich die Packer befinden, "", wenn im aktuellen Pfad oder im Suchpfad zu finden
`{}`	wiederholen für alle Testdateien
`<name>`	Name der Testdatei
`<bezeichnung>`	Erklärung
`<extension>`	Kurzcharakterisierung (erscheint in den Spalten der Tabelle)

Beispiel

```
D:
\pcn\4x\43\zip\test
```

N42ASC.TXT	ASCII-TEXT	TXT	379.951
N42A.DOC	WINWORD-TEXT	DOC	1.164.288
PCN.MDB	ACCESS-Daten	MDB	6.488.064
MULTIMAN.TIF	TIF-Datei	TIF	1.868.258
WINWORD.EXE	EXE-Datei	EXE	3.482.624
PCK.PCK	Gepackte-Datei	PCK	2.039.057

Die Datei `PCK.PCK` ist die mit `PKZIP` gepackte Datei `WINWORD.EXE`. Sie soll zeigen, ob es ein Verfahren gibt, welches bereits gepackte Dateien schneller archivieren kann.

Die letzte Spalte (Dateigröße) muß nicht eingegeben werden, das Programm bestimmt die Dateigröße selbst.

Ergebnisse

Da die Dateien unterschiedlich lang sind, wurde in der Ergebnistabelle jeweils auf 1 MB umgerechnete Zeiten eingesetzt. Alle Programme wurden ohne weitere Optionen gestartet, obwohl durch einige Detailkenntnis auch noch einiges herauszuholen wäre, was aber dem experimentierfreudigen Leser überlassen bleiben soll. Achtung: die gemessenen Zeiten können bei verschiedenen Durchläufen durchaus schwanken.

		Version	TXT			DOC			MDB			TIF			EXE			PCK		
	Zeiten/MB(s)/Packverhältnis		pack	xpack	%	pack	xpack	%	pack	xpack	%	pack	xpack	%	pack	xpack	%	pack	xpack	%
A	GZIP	1.1.1	18.3	6.34	36	12.5	4.58	27	14.6	4.85	34	8.44	2.5	8	19.6	7.02	58	18.3	7.06	100
B	PKZIP/PKUNZIP	2.04g	11.4	5.18	36	9.25	2.97	27	8.62	2.26	34	6.91	2.09	8	12.3	2.64	58	14.4	2.29	100
C	LHA	2.13	20.5	4.47	38	18.8	2.93	28	16.2	3.08	37	16.6	1.76	7	17.5	4.08	60	18.8	2.96	100
D	ZOO	2.1	14.6	7.37	51	12.5	6	38	14.6	6.25	52	7.67	5.14	10	17.9	7.95	78	34.4	5.23	100
E	ARJ	2.41a	20.5	5.34	36	15.7	3.31	27	17.3	2.67	34	16.3	2	8	17.9	3.5	58	8.29	4.18	100
F	PKARC/PKXARC	3.5	6.95	4.47	51	4.29	2.64	38	4.61	2.4	52	2.85	1.85	10	5.96	2.76	78	10.6	2.94	100
G	PAK	2.51	15.2	5.63	39	21.7	3.73	34	14.3	3.32	41	9.93	2.79	10	14.2	4.29	61	2.88	3.02	100
	Sieger		F	C,F	A,B,E	F	F	A,B,E	F	B	A,B,E	F	C	C	F	B	A,B,E	G	B	-
	Verlierer		C,E	D	D,F	G	A	D,F	C	D	D,F	C,E	D	D,F,G	C,D,E	A,D	D,F	D	A	-

Interpretation

Der Vergleich zeigt, daß Texte, Tabellen, Datenbanken und auch EXE-Dateien weitgehend ähnliche Kompressionsergebnisse liefern. Lediglich bei Bilddateien und bei bereits komprimierten Dateien ergeben sich Unterschiede.

	Sieger	Verlierer
Packzeit	PKARC	LHA
Entpackzeit	PKZIP	ZOO
Packverhältnis	GZIP,PKZIP,ARJ	ZOO,PKARC
Packzeit gepackte Datei	PAK	ZOO
Entpackzeit gepackte Datei	PKUNZIP	GZIP

Ginge es nur um die Kompressionszeiten oder um das Kompressionsverhältnis, kann man je nach Kriterium dem einen oder anderen Programm den Vorzug geben.

Man sollte aber auch die durch die Programme angebotenen sonstigen Eigenschaften vergleichen. Beispielsweise:

- selbstentpackende Archive
- rekursives Archivieren von Verzeichnissen
- Archivieren großer Dateien auf mehrere Disketten, Backupfunktion
- Paßwort
- Benutzeroberfläche oder Kommandozeile
- Reparieren von Archiven

Empfehlung

Sehr empfehlenswert ist das Programm PKZIP/PKUNZIP. Es ist nicht das schnellste beim Packen aber auch nicht das langamste, dafür entpackt es sehr schnell und hat darüberhinaus im Schnitt das vorteilhafteste Packverhältnis. Außerdem erfordert PKZIP/PKUNZIP für den einfachsten Fall keine sonstigen Optionen.

Weitere Kriterien

	Kommandozahl	Preis
ARJ	160	$ 40,-
GZIP	13	0,-
LHA	28	0,-
PAK	45	$15/30/75
PKZIP/PKUNZIP	41/23	$ 47,-
PKARC/PKXARC	13/8	$45,-
ZOO	51	0,-

Literatur

PCC-Magazin 29, März/April 1995, Seite 15.

Das Testprogramm (PACKTST.CPP)

Das Testprogramm wurde als C++-Programm konzipiert. Es wurden einige Eigenschaften von C++ mitaufgenommen, die die Handhabung sehr erleichtern. Das komplette Programm wird hier aus Platzgründen nicht abgedruckt. Einige Merkmale werden hier beschrieben.

Die einzelnen Packer werden ausgeführt, indem der String commandline für alle Packer und für alle Testdateien mit der Funktion `system()` ausgeführt wird. Für die Zeitmessung wird die unten dargestellte Funktion `counter()` verwendet. Das Ergebnis wird in die Struktur `testresult[][]` geschrieben, die zur Laufzeit gemäß dem Produk aus Anzahl der Packer `packcnt` und Anzahl der Testdateien `testcnt` gebildet wird.

Am Ende des Tests werden die Werte in diesem zweidimensionalen Array auf jeweils 1MB Datenmenge normiert.

`Streams`

Für Ausgabe am Bildschirm und in Datei. An Hand des selbstdefinerten `ofstream rout` wird gezeigt, wie man vorgeht.

`Cstring`

Die Verarbeitung von Strings in C ist sehr fehleranfällig, da man selbst für ausreichenden Speicherplatz sorgen muß. Nicht so in C++. Verwendet wurde eine allgemeine String-Klasse der Microsoft-Foundation-Classes `Cstring`, mit der Strings auf einfache Weise verkettet werden können, ohne den jeweiligen Platzbedarf im vorhinein abschätzen zu müssen.

`counter`

```
enum COUNTERMODE { CTR_START, CTR_STOP, CTR_STOPREAD, CTR_SUSPEND, CTR_RESUME};

double counter(enum COUNTERMODE mode)
{
  static clock_t start;
  clock_t finish;
  static double duration;

  switch (mode)
  {
  default: return duration;
  case CTR_START:
    start = clock();
    duration = 0.0;
    return duration;
  case CTR_STOP:
    finish = clock();
    duration = duration + (double)(finish - start) / CLOCKS_PER_SEC;
    return duration;
  case CTR_STOPREAD:
    finish = clock();
    duration = duration + (double)(finish - start) / CLOCKS_PER_SEC;
    printf( "\n%f Sekunden\n", duration );
    return duration;
  case CTR_SUSPEND:
    finish = clock();
    duration = duration + (double)(finish - start) / CLOCKS_PER_SEC;
    return duration;
  case CTR_RESUME:
    start = clock();
    return duration;
  }
}
```

Das Programm counter wird an jenen Stellen des Programms aufgerufen, wo man einen Timer startet (`counter(CTR_START)`) und die Zeitmessung beginnen soll und wo man ihn wieder ablesen will `time= counter(CTR_STOP)`. Man kann die Zeitmessung unterbrechen `(CTR_SUSPEND)` und wieder starten `(CTR_RESUME)`.

Es gibt auch Packer für Windows 95. WINZIP-95 kann auch schon lange Dateinamen verwalten. Sie finden das Programm auf Diskette 498. □